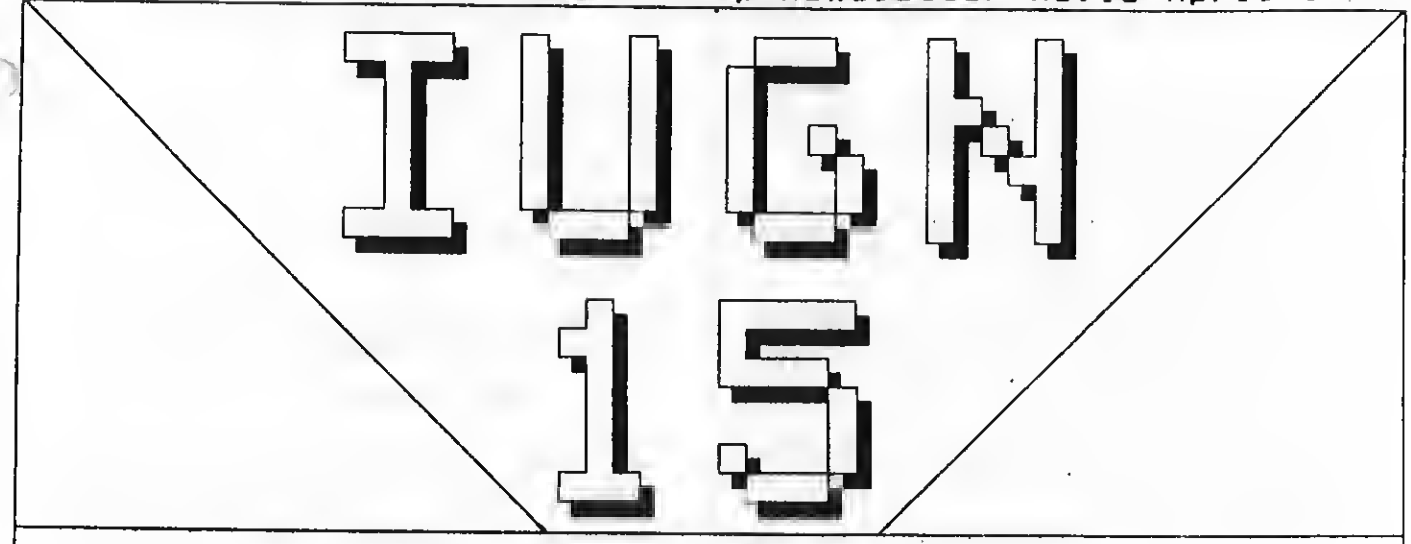


Interak User Group Newsletter No.15 April 1987

CONTENTB

DATA PROTECTION ACT 1985:
Detaila of the membera of the IUGN are held on computer file. Each member may view, alter or deetroy any data held on him/her within that file. To obtain a cooy of your file please aend a atamped addreaaed envelope to the Editor. Your eubaequent wiahea regarding that data will be honoured without queetion.

NOTES

The disk software library is now fully operational and you will find at the rear of this issue the first pages of the Interak User Group Index to the Public Domain Disk Software Library.

This has been designed so that you can collect the pages, one per disk volume, into a seperate folder. Updated and additional pages will be issued as and when required.

It is recommend that you get a special folder to place these pages in. It will act as an ever growing reference book to the public domain software supplied by the user group.

By adopting one page per volume, the librarian can re-issue a volume page to include any information gained on that volume since it's first issue.

Remember if you have any gems for the "Public Domain" send them to Charlie who will make them available to everbody. Fame is within your reach.

Your Disk Librarian is :-

Mr C.V.Bridgstock, 32 Wimborne ave, Thingwall, Wirral, Merseyside, L61 7UL. Phone 051-648-3888.

A new VDU-2K character generator Eprom is available from Greenbank, order as CB2V020B. It alters the set to be true ASCII.
Changes are :-

Exclamation mark display improved.
Pound sign changed to be "\"
Divide sign changed to be "|"
Has Chessmen for Disk Library version of CHESS643.COM.

A new card has been released by Greenbank. It is called a PRN-3 and is a parallel printer interface. It will connect a printer to the Interak Isbus. Details from Greenbank.

CP/M plus is now available. It needs a 64k Ram capable system as a minimum. It will let you expand your system past the 64k limit of CP/M 2.2. CP/M plus is sometimes called CP/M 3.0. Contact Greenbank for details.

If you can, please send items for inclusion in the newsletter "on-disk". I will copy the disk and return your original as fast as possible. For you it gives the greatest chance that I don't introduce errors during the re-type. For me it saves on finger wear.

David has mentioned to me that he is working on an 80 column VDU design. He has chosen the driver chip and has reached the point of trying to fit it all on a new card. I think that this is the best Interak related news I have heard for agas. The 64 column screen is only just tolerable with disks and inhibits many software pakages. I wish David well in the design and look forward to the day of it's release.

If you are held back from installing your 3.5" disk drives due to the lack of a mounting kit you may like to try this idea in the short term. Get an old unwanted card and cut 2" off of it at the connector end. Mount the disk onto the card and slide it into the rack. You can prevent the assembly from moving backwards and forwards by a sticky tape wrap on the card guides and it will quite happily chug away until the mounting kit is received. If you don't have an old card any type of rigid board will suffice.

Bob Eldridge.

WHERE TO BUY CP/M SOFTWARE

CP/M v2.2 and v3.0 Disk operating system.
Greenbank Electronics, 460 New Chester road, Rock Ferry, Birkenhead, Merseyside, L42 2AE.
051-645-3391.

"C" language Compilers and Interpreters.
Grey Matter Ltd, 4 Prigg Meadow, Ashburton, Devon, TQ13 7DF.0364-53499.

ZORK,1,2 and 3. Sophisticated adventure games.
Anita Business Systems Ltd, London. 01-253-2444.

MEMBERS TAPE SOFTWARE - FOR SALE

You may use this section to sell tape software to other users. Send a brief description of your product giving details of its distribution and price, to the EDITOR. Note that you will be responsible for the support of your own product. See CONTACTS for "ORDER FROM" addresses. Software supplied is the responsibility of the "ORDER FROM". Please deal directly with the "ORDER FROM" in the event of bugs ect.

MACHINE CODE

NAME	DESCRIPTION	ORDER FROM:	COST
FIGFORTH	FORTH COMPILER	D.CAMBELL	£15.00
INTERPLAY	BB DRIVER	M&M ELECT	£ 4.00
MEDABUG	DEBUGGER	P.VELLA	£13.00
VELTEXT	TEXT EDITOR	P.VELLA	£ 5.00
XTAL BABIC	14K BASIC	P.VELLA	£40.00
ZYBASIC 3A	(DK TAPE)	GREENBANK	£15.95
ZYBASIC 3C	(IN ROM)	GREENBANK	£27.75
ZYMON 2.V203	MONITOR PROGRAM	GREENBANK	£15.95

XTAL BASIC

NAME	DESCRIPTION	ORDER FROM:	COST
AWARI	GAME	M.SAUNDERS	PP
BIORYTHMS		M.SAUNDERS	PP
CHAR DES	CHARACTER BUILDER	M.SAUNDERS	£ 5.50
I-SPY	GAME	M.SAUNDERS	PP
SOUND DEV	SOUND DEVELOPMENT	M.SAUNDERS	£ 5.50

Key: PP = Postage & packing.
POA = Please enquire (Phone for price.)

CONTACTS

BACK ISSUES... Can be obtained from:-
D.Parkins, Greenbank Electronics, 460 New Chester road, Rock Ferry, Birkenhead, Merseyside, L42 2AE.

BOOKS..... Lend, borrow, and swop books via :-
R.E.Bowyer, 45 Ford drive, Yarnfield, Stone, Staffs.

D.CAMBELL .... 153 Lower Fairmead road, Yeovil, Somerset, BA21 5SR. Tel 0935-702B2.

DISK LIBRARY . Public domain disk software from :-
Mr C.V.Bridgstock, 32 Wimborne ave, Thingwall, Wirral, Mersayside, L61 7UL. Phone 051-64B-3888.

EDITOR..... Send submissions to :-
R.Eldridge, 2B Wycherley Close, Blackheath,London,SE3 7QH.

GREENBANK .... D.Parkins, Greenbank Electronics, 460 New Chester road, Rock Ferry, Birkenhead, Merseyside, L42 2AE.

M&M ELECT .... 8 Ayra view, Bride, Isle of man.

M.SAUNDERS ... M.Saunders, 7 Drumcliff road, Thurnby Lodge,Leicester,LE5 2LH.

MEMBERSHIP.... To join, renew or change your details contact :-
Tom Evans, 129 Cranbourne Waye, Hayes, Middlesex, UB4 0HR.

P.VELLA ..... 19 Ford Drive, Yarnfield, Staffs.

SUBSCRIPTIONS. For information and payments please contact:-
Tom Evans, 129 Cranbourne Waye, Hayes, Middlesex, UB4 0HR.

WORDSTAR. Command sequences

```
SIGN ON MENU
D    Edit a document file.
E    Rename a file.
F    Switch directory listing on/off.
H    Set help level.
L    Change logged disk drive.
M    Merge-print optional.
N    Edit a Non-document file.
O    Copy a file.
P    Print a file.
R    Run a program. Com file.
X    Exit to CP/M.
Y    Delete a file.

 ^ means press the CTRL key, and type the letter.
CURSOR
^A   Cursor left one word.
^F   Curor  right one word.
^S   Cursor left one character.
^H   Cursor left one character.
^D   Cursor right one character.
^E   Cursor up one line.
^X   Cursor down one line.
^QS  Cursor to start of line left side.
^QD  Cursor to end of line right side.
^QE  Cursor to top of screen.
^QX  Cursor to bottom of screen.
^QR  Cursor to start of file.
^QC  Cursor to end of file.
^QB  Cursor to start of marked block.
^QK  Cursor to end of marked block.
^QP  Cursor to previous position.
^QV  Cursor source.
^Q0-^Q9  Cursor to marker 0-9.

SCROLL
^C   Scroll up one screen.
^R   Scroll down one screen.
^W   Scroll down one line.
^Z   Scroll up one line.
^QW  Scroll down continuously.
^QZ  Scroll up continuously.

DELETE
^Y     Delete line.
^T     Delete word right.
^G     Delete character right.
^KJ    Delete additional file.
^KY    Delete marked block.
^QY    Delete to the end of the line.
^Qdel  Delete to start of line.
DEL    Delete character left.

TABS AND MARGINS
TAB  Tab.
^I   Tab
^QO  Paragraph tab.
^ON  Clear tab stop.
^OI  Set tab stop.
^OL  Set left margin.
^OR  Set right margin.
^OX  Release margin.
^OF  Take margins and tabs from line.

SWITCHES
^KF  File directory on/off.
^OD  Display print controls on/off
^OH  Hyphen help on/off.
^OJ  Right justification on/off.
^OT  Ruler display on/off.
^OV  Variable tabs on/off.
^OW  Word wrap on/off.
^OP  Page break display on/off
^V   Insert on/off.

CARRIAGE RETURNS
RETURN Insert a carriage return, cursor down.
^M     Insert a carriage return, cursor still.
^N     Insert a carriage return, cursor still.

FIND/REPLACE
^L   Find/replace again.
^QA  Relace
^QF  Find
^Q9  Repeat next command.

SET
^OI  Set tab stop.
^OL  Set left margin.
^OR  Set right margin.
^JH  Set help level. 0,1,2,3.
^OS  Set line spacing.
^K0-9 Set/hide marker 0-9

BLOCK
^KB  Mark block beginning.
^KK  Mark block end.
^KC  Copy marked block block.
^KV  Move marked block.
^KH  Hide/display marked block.
^KW  Write marked block to additional file.

DISK DRIVE
^KL  Change logged disk.

ADDITIONAL FILE
^KE  Rename additional file.
^KO  Copy additional file.
^KP  Print additional file.
^KR  Read additional file.
^KW  Write marked block to additional file.

SAVE OPTIONS
^KD  Finished edit. Save and return to main menu.
^KQ  Abandon edit.
^KS  Save and continue editing.
^KX  Save edited file and exit to CP/M.

MISC
^B         Reform the paragraph.
^U         Interrupt current command.
^OC        Center text on line.
^OE        Soft hyphen entry.
^PA-^PZ    Enter ^A-^Z into text.
^PM        Make next line overprint last line.
^PO        Enter non-break space.
ESCAPE     Error release.
LINE FEED Same as ^J.

EXPLAIN THINGS
^J1  Explain menu pages and text entry
^JB  Explain paragraph reform works.
^JD  Explain print directives.
^JF  Explain how flags work.
^JH  Explain tabs and margins.
^JP  Explain place markers.
^JR  Explain the ruler line.
^JS  Explain status line.
^JV  Explain moving text.

DOT COMMANDS
.LH  Line hight
.PL  Paper length
.MT  Margin at top
.MB  Margin at bottom
.HM  Heading margin
.FM  Footing margin, (page & margin)
.PC  Page & column
.PO  Page offset
.PA  New page
.CP  Conditional page
.HE  Heading
.FO  Footing
.OP  Omit page numbers
.PN  Page number
.CW  Character width
.SR  Subscript roll
.UJ  Micro-justify
.DP  Bidirect print
.IG  Comment
..   Comment
```

Evans above

MODEM MANIA!

The Taecomm Bulletin Board has been running with a WS4000 modem for a couple of months now, giving our users a choice of speeds, V21 300/300 and V23 1200/75, these are auto selected by the modem when it recaivas the call, it then evaluates the carrier tone, and tells the Interak to change the COM1 card to suit, pretty nifty eh? Anyway no problems with the modem es yet, a couple of users had problems, but this turned out to be the way thay had set up their modems, end software, needless to say we managed to find the individual problems between us, and sort things out. The software that runs the board, was reasonably easy to change to suit the new modem, but the modem control segment is radically differant running with the "Hayes" commands compared with the old Datel 28 modem, and no alterations to the COM1 card were needed, just pulled the plug on tha old 28, and plugged in the new WS4000, that made life a bit easier for me. The only real problem encountered in re-writing the softwara, was that the return codes from the modam are issued in escii instaad of hex (ie. 38h instead of 08h), tha modem manual did not warn about this "featura", therefora making things very frustrating whan testing, it took me two days of mucking around befora I eventually twigged what I was doing wrong (yes I know I'm dim). Apart from that little problem, its a pratty good little buy, and I understand that Greenbank is also stocking this model, so if you fancy one, splash out the ackers.

SOON (ish)!

I will hopefully be adding an extra SIG section to the Taecomm Board (if we get anough Interakers using it), this will be dedicated to Interak users only, and will be accessed by a coda that will be included in the user log, also for those lucky enough to have a colour display, and the software support, we will be adding "Prestel" compatible type graphics and menus. Telecom has supplied me with a software guide to protocol, and text graphics, so I will be "trying" to get this running on the board as soon as possible, but no promises on that one, cos a lot of long winded testing will be needed, and depends upon the goodwill of somaone actually calling the board to do repeated (real pain, and expensive) tests of tha software. If anyone has any knowledge on the subject of comms graphics, let me know, I can do with all the help I can get on this one.

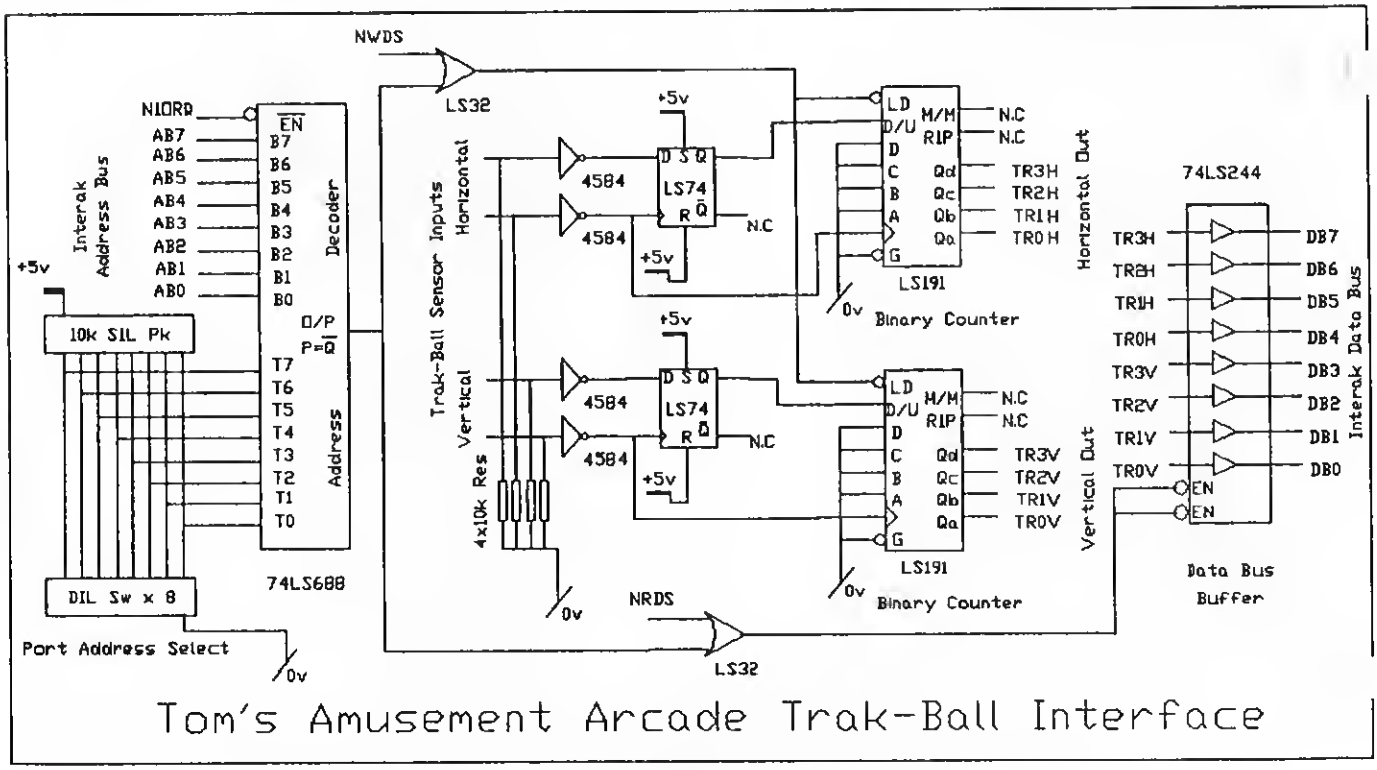
LURKING CLONE

Furtivaly lurking in amongst the Intareks is a lone IBM clone that is slowly revealing its hidden powers es more software is becoming available to me, It is rapidly becoming a very valuable tool as I become mora used to the operating system (PCDOS/MSDOS), and is very similar to CP/M but much more forgiving. I heve been using it mainly for graphics at the presant, and since I fitted the 8087 maths co-processor, it madly calculates all graphic construction et a speed far beyond the original configuration of the mechine. The most recent application was the scaled graphic re-construction of room, and furniture fittings, to my surprise everything actually fitted when assembled in real life! It is a pity that this type of software is not available for CP/M type systems. It should be relatively aasy es an excercise to implement the 8088/8086 + MSDOS to the Interak bus without too much expense. MSDOS oparating system and manuals are available now for as little as 50 quid or thereabouts, this will give access to a very large amount of modern softwara, and a huge public domain sector of softwa e comparible to CP/M, but very much up to date. This software is also compatible with the later 32 bit versions of the same stable processors, that are coming into the market. ANYONE INTERESTED IN IBM COMPATIBILITY??

ARCADE TRAK-BALL

Below is e circuit diagrem for intarfacing a Trak-Ball to your beloved Interak, Trak-Ball used in this case wes an ex-Atari pub game, complete with fungus from old beer droppings (yuk!), this I obtained from a local arcade game maintenanca chappy for 15 quid, and it is very tough with real ball bearings, and fings. The circuit is tried and tested, all you need is a little bit of software to read it, and that is available if required.

Sees yas..Tom.



Tom's Amusement Arcade Trak-Ball Interface

PART SCREEN DRIVER FOR THE VDUK/2K

By R.A.Cowdery

The following is a brief description of a part screen driver for the VDUK, there is no reason why it should not also work on the VDU2K, though as I do not posses one this has not been tested (should only be necessary to change the equate WIDTH to 64). The routines allow a number of part screens to be opened, at present limited to 10 by the control blocks allocated in the program (easily changed). Once opened each part screen (which may be of any size from 2 characters to a full screen), may be written to independantly by quoting the stream number in the call (in the B register). Each part screen will scroll (horizontally if the screen is a single row), the cursor may be selectively turned on or off for each part screen, and each may be either in graphics or text mode. Certain control codes are implemented as detailed below if the screen is in text mode, else all characters are treated as printable. A small test routine is given at the end by means of a demonstration of some of the capabilities.

In case there are any Z80 boffins out there, I would mention that this is the first Z80 assembler I have written and it may therefore be a bit cluesy in places. I must admit I found the instruction set difficult to get to grips with. There seemed to be so many instructions and yet never the one that I wanted to do the job. All suggestions welcome !!!

There is reasonable error checking in the routines, though I would'nt claim them to be uncrashable. Some of the control codes used are non-standard, however they may be easily changed by changing the table CONTAB.

The interfaces are very simple, consisting of calls to the following :-

```
OPEN(X,Y,X',Y') X = start column . passed in B reg
                Y = start row .... passed in C reg
                X'= end column ... passed in D reg
                Y'= end row ..... passed in E reg
```

Note that the coordinates delimit the part screen and are inclusive of the columns and rows specified. There is nothing that prevents the part screens being overlapped and some interesting results will be obtained if you do this (not useful, just interesting!). Seriously, provided some disipline is used this deficiency can be useful, eg you could open a largish part screen which can have titles written into it, borders for smaller part screens and can be cleared in a single call etc.

```
Error code returned in A reg :-
00H - Open successful
F9H - A single character screen has been defined.
FAH - Part screen defined is > max no. of rows.
FBH - Part screen defined is > max no. of cols.
FCH - Y > Y'
FDH - X > X'
FFH - No free streams
Stream returned in B reg, valid only if A=00H
```

```
CLOSE(Stream)  Stream = stream returned in open.
                       passed in B reg.
```

```
Error code returned in A reg :-
00H - Close successful
0EH - Stream specified is > max stream.
0FH - The stream specified was not currently open.
```

```
WRITE(Char,Stream,[X,Y])

          Char   = Character to write.
                   Passed in A reg.
          Stream = Stream returned in OPEN.
                   Passed in B reg.
          [X,Y]  = Cursor coordinates for
                   move cursor command.
                   Passed in D,E regs.
```

The character may be any character code, if the mode is graphics (selectable by character code 12H) then the character will simply be printed else certain control characters are available as detailed below :-

```
Clear part screen - 0CH
Carriage return   - 0DH
Line feed         - 0AH
Cursor south      - 0AH
Backspace         - 08H
Cursor west       - 08H
Horizontal tab    - 09H
Cursor east       - 09H
Vertical tab      - 08H
Cursor north      - 08H
Delete (destructive backspace) - 7FH
```

The following are additional functions using DC codes, some of these are non-standard :-

```
Cursor toggle     - 11H
Mode toggle       - 12H
Home cursor       - 13H
Position cursor   - 14H
```

NB each call to toggle eg the cursor will invert the state.

Error codes returned in A reg :-

```
00H - Write successful
EAH - For position cursor, cursor Y coord is
      outside part screen.
EBH - For position cursor, cursor X coord is
      outside part screen.
EEH - The stream specified is > max part screens.
EFH - The stream specified is not currently open.
```

```
;*****************************************
;       VDUK/VDU2K PART SCREEN DRIVER
;       Author R.A.Cowdery
;       Date   6th June 1986
;       Issue  1
;*****************************************
;DESCRIPTION
;   This set of routines enables up to a (user
;   definable) number of part screens to be opened
;   and written to
;
; INTERFACES:  OPEN
;              CLOSE
;              WRITE

            ORG 0A000H
            LOAD 0A000H
;
;General Equates :
NOTALL:     EQU  0FFH    ;SCB NOT ALLOCATED
MXPT:       EQU  10      ;MAX PART SCREENS
SHIFT:      EQU  5       ;MULT BY 32
BASE:       EQU  0F000H  ;ROW 1 ADDRESS OF SCREEN
WIDTH:      EQU  32      ;SCREEN WIDTH FOR VDUK
DEPTH:      EQU  24      ;SCREEN DEPTH
SCBSIZE:    EQU  8       ;SIZE OF SCREEN CONTROL
                         ;BLOCKS
WKSSIZE:    EQU  15      ;SIZE OF WORKSPACE BLOCK
MAXCONT:    EQU  15      ;MAX CONTROL CODE USED
;
; ERROR CODES
; ***OPEN ERRORS***
NOSRT:      EQU 0FFH     ;ALL STREAMS CURRENTLY
                         ;OPENED
XORT:       EQU 0FDH     ;CANT FINISH BEFORE YOU
```

```
                        ;START
YORT:       EQU BFCN    ;DITTO FOR Y
COLTOBIB:   EQU BFBH    ;PART SCREEN LARGER THAN
                        ;MAX COLS
ROWTOBID:   EQU BFAN    ;DITTO FOR ROWS
SINGLECHAR: EQU BF9H    ;A SINGLE CHAR IS NOT A
                        ;VALID SCRN

; ***WRITE ERRORS***
STRNOTOPEN: EQU BEFH    ;THE STREAM SPECIFIED IS
                        ;NOT OPEN
STRTOBIDW:  EQU BEEH    ;STREAM >15
XCURINV:    EQU BEBH    ;FOR POSITION CUR,X IS
                        ;OUTSIDE SCR
YCURINV:    EQU BEAH    ;DITTO FOR Y

; ***CLOSE ERRORS***
STRNOTOPNC: EQU BOFH    ;THE STREAM WAS NOT OPEN
STRTOBIDC:  EQU BDEH    ;STREAM>15

SUCCESS:    EQU 88      ; ***SUCCESS CODE***

;OFFSETS INTO SCB
SCBSTR:     EQU  0      ;STREAM NUMBER, FFH WHEN
                        ;UNALLOC
SCBSSH:     EQU  1      ;X COORDINATE (COL)
SCBSSV:     EQU  2      ;Y COORDINATE (ROW)
SCBSEH:     EQU  3      ;X' COORDINATE (COL')
SCBSEV:     EQU  4      ;Y' COORDINATE (ROW')
SCBCUX:     EQU  5      ;CURSOR X COORDINATE
SCBCUY:     EQU  6      ;CURSOR Y COORDINATE
SCBATT:     EQU  7      ;STATUS WORD

;BIT DEFINITIONS FOR SCBATT
SCBBDR:     EQU  0      ;BIT 0, NORMAL=0,
                        ;       GRAPHICS=1
SCBBCU:     EQU  1      ;BIT 1, CURSOR ON=0,
                        ;       CURSOR OFF=1
;STORAGE ALLOCATIONS
WKSP:       DS   WKSSIZE ;WORKSPACE BLOCK:

SCB1:       DB NOTALL   ;SCREEN CONTROL BLOCKS
            DS SCBSIZE-1
SCB2:       DB NOTALL
            DS SCBSIZE-1
SCB3:       DB NOTALL
            DS SCBSIZE-1
SCB4:       DB NOTALL
            DS SCBSIZE-1
SCB5:       DB NOTALL
            DS SCBSIZE-1
SCB6:       DB NOTALL
            DS SCBSIZE-1
SCB7:       DB NOTALL
            DS SCBSIZE-1
SCB8:       DB NOTALL
            DS SCBSIZE-1
SCB9:       DB NOTALL
            DS SCBSIZE-1
SCB10:      DB NOTALL
            DS SCBSIZE-1

SCBL:       DW SCB1     ;SCREEN CONTROL BLOCK LIST
            DW SCB2
            DW SCB3
            DW SCB4
            DW SCB5
            DW SCB6
            DW SCB7
            DW SCB8
            DW SCB9
            DW SCB10

CONTROLTAB: DW ENDCONT  ;CONTROL TABLE
            DW ENDCONT
            DW ENDCONT
            DW ENDCONT
            DW ENDCONT
            DW ENDCONT
            DW ENDCONT
            DW ENDCONT
            DW BS       ;BACKSPACE
            DW HT       ;HORIZONTAL TAB
            DW LF       ;LINEFEED
            DW VT       ;VERTICAL TAB
            DW CS       ;CLEAR SCREEN
            DW CR       ;CARRIAGE RETURN
            DW ENDCONT
            DW ENDCONT
            DW ENDCONT
            DW ENDCONT
            DW CT       ;CURSOR TOGGLE
            DW NT       ;NOOP TOGGLE
            DW HC       ;HOME CURSOR
            DW PC       ;POSITION CURSOR
;*********************************************
; OPEN (X,Y,X',Y')
; FUNCTION: This is the main control routine of
;           the OPEN function.
; INPUTS : B=Start col X
;          C=Start row Y
;          D=End col X'
;          E=End row Y'
; OUTPUTS: B=Stream number
;*********************************************
OPEN:LD IX,WKSP   ;SET UP WORKSPACE
     LD (IX+0),B  ;COL
     LD (IX+1),C  ;ROW
     LD (IX+2),D  ;COL'
     LD (IX+3),E  ;ROW'
                  ; ***OPEN ERROR CHECKS***
     LD A,D       ;X'
     CP B         ;X'-X
     JP M,ERR1    ;X>X'
     LD A,E       ;Y'
     CP C         ;Y'-Y
     JP M,ERR2    ;Y>Y'
     LD A,WIDTH
     DEC A
     CP D         ;WIDTH-X'
     JP M,ERR3    ;X'>WIDTH
     LD A,DEPTH
     DEC A
     CP E         ;DEPTH-Y'
     JP M,ERR4    ;Y'>DEPTH
     LD A,B       ;X
     CP D         ;X'-X
     JP NZ,CONT   ;X<>X'
     LD A,C
     CP E         ;Y'-Y
     JP NZ,CONT
     JP ERR5      ;SINGLE CHAR SCREEN
CONT:CALL ALLOCATE;ALLOCATE A FREE SCB IF POSSIBLE
     CP 0FFH      ;ALLOCATION SUCCESSFUL
     JP Z,EXIT    ;ALL STREAMS ALLOCATED
     PUSH AF
     CALL INIT    ;INITIALISE ALL FIELDS IN SCB
     CALL CLEAR   ;CLEAR THE ALLOCATED PART SCREEN
     CALL SCURSOR ;TURN ON THE CURSOR AT RELATIVE
                  ;B,B
     POP BC       ;STREAM NUMBER IN B
     LD A,SUCCESS ;SUCCESSFUL EXIT
     JP EXIT
ERR1:LD A,XORT
     JP EXIT
ERR2:LD A,YORT
     JP EXIT
ERR3:LD A,COLTOBID
     JP EXIT
ERR4:LD A,ROWTOBIO
     JP EXIT
ERR5:LD A,SINGLECHAR
     JP EXIT
EXIT:RET          ;OPEN COMPLETED
     DB 0FFH
;*********************************************
; ALLOCATE ()
; FUNCTION: Allocates a fre SCB from the pool of
;           SCB's
; INPUTS : NONE
; OUTPUTS: A=Stream number (1-15), or FFH if no
;          free SCB's
;          HL = Pointer to start of allocated SCB
; LOGIC  : FOR n=1to15
;          DO
;            pointer=(SCBL+n)
;            IF pointer!SCBSTR<>FFH THEN
;              BREAK
;            ELSE
;          OD
;*********************************************
ALLOCATE:LD B,88      ;LOOP COUNTER
         LD IY,SCBL-2 ;BASE OF TABLE -2, ALLOW
                      ; FIRST INC
```

```
ALLOOP:  INC IY          ;STEP TO NEXT
         INC IY          ;OR FIRST ENTRY
         INC B           ;BUMP LOOP COUNTER, ALSO
                         ; STREAM NUMBER
         LD L,(IY+0)     ;ADR OF SCB AT STREAM NUMBER
         LD H,(IY+1)     ;IN HL
         LD A,(HL)       ;SCBSTR IN A
         CP 0FFH         ;THIS SCB ALLOCATED?
                         ; SCBSTR=FFH IF FREE
         JP Z,FOUND      ;NO, SO CLAIM IT
         LD A,B          ;CHECK FOR END OF SCB'S
         CP MXPT         ;END?
         JP NZ,ALLOOP    ;NO, SO TRY NEXT SCB
         LD A,0FFH
         RET             ;FAIL EXIT
FOUND:   LD A,B          ;STREAM NUMBER RETURNED IN A
         RET             ;SUCCESSFUL EXIT
;*************************************************
; INIT (SCB POINTER,STREAM NUMBER)
; FUNCTION: Initialise all fields in the allocated
;           SCB
; INPUTS:  A = Stream number (1-10)
;          NL= Pointer to allocated SCB
; OUTPUTS: None
;*************************************************
 INIT:PUSH NL
      POP IY             ;MAKES INDEXING EASIER
      LD B,(IX+0)        ;RESTORE REGS FROM WORKSPACE
      LD C,(IX+1)
      LD D,(IX+2)
      LD E,(IX+3)
      LD (IY+SCBSTR),A   ;STREAM NUMBER
      LD (IY+SCBBBN),B   ;X
      LD (IY+SCBBBV),C   ;Y
      LD (IY+SCBBEN),D   ;X'
      LD (IY+SCBBEV),E   ;Y'
      LD (IY+SCBCUX),B   ;INIT CURSOR POSITION NOME
      LD (IY+SCBCUY),C
      LD (IY+SCBATI),00  ;CURSOR ON, NORMAL MODE
      POP HL             ;LEAVE TIDY
      RET
;*************************************************
; CLEAR (WORKSPACE)
; FUNCTION: Clear the allocated part screen to spaces
; INPUTS:  WKS0 = START COL (X)
;          WKS1 = START ROW (Y)
;          WKS2 = END ROW (X')
;          WKS3 = END COL (Y')
; OUTPUTS: NONE
; WORKSPACE USAGE: WKS4,5 = BASE+(Y*WIDTH)+X
;                  WKS6   = (Y'-Y)+1
;                  WKS7   = (X'-X)+1
;                  WKS8   = WIDTH-(X'-X)
; LOGIC:   Set up workspace
;          Start = WKS4,5
;          FOR n = 1 TO WKS6
;          DO
;            FOR m = 1 TO WKS7
;            DO
;              [start] := space
;              inc start
;            OD
;            start = start + WKS8
;          OD
;*************************************************
CLEAR:  CALL SETUPWKS45 ; SET UP WORKSPACE
        CALL SETUPWKS6
        CALL SETUPWKS7
        CALL SETUPWKS8
        LD N,(IX+4)  ; START = WKS4,5
        LD L,(IX+5)  ;HL = BASE + (Y*WIDTH) + X =
                     ;START
                     ;FOR N = 1 TO WKS6
OUTER:  LD B,(IX+6)  ;OUTER LOOP COUNT IN B, NUMBER
                     ;LINES
                     ; FOR M = 1 TO WKS7
        LD C,(IX+7)  ;INNER LOOP COUNT IN C,
                     ;NUMBER COLS
; [START] := SPACE
INNER:  LD (NL),20H  ;CLEAR CELL
; START = START + 1
        INC NL       ;NEXT CELL TO CLEAR
        DEC C        ;LAST LINE CELL OF THIS PART
                     ;SCREEN
        JP NZ,INNER  ;NO
; START = START + WKS8
        DEC B        ;ALL LINES OF THIS PART SCREEN
                     ;CLEARED?
        JP Z,DONE    ;YES
        LD (IX+6),B  ;SAVE OUTER LOOP COUNT
        LD C,(IX+8)  ;C = WIDTH-(X'-X)
        LD B,00      ;READY FOR ADD
        ADD NL,BC    ;NL = START + WKS8
        JP OUTER     ;CLEAR NEXT LINE
DONE:   RET          ;ALL CLEARED
;
; SET UP WORKSPACE 4,5
SETUPWKS45:
        LD L,(IX+1)  ;Y
        CALL ROWNO   ;CALC WIDTH*Y
        LD BC,BASE   ;SCREEN BASE ADDRESS
        ADD HL,BC    ;NL = BASE + (WIDTH*Y)
        LD C,(IX+0)  ;X
        LD B,00      ;READY FOR ADD
        ADD NL,BC    ;HL = BASE + (WIDTH*Y) + X
        LD (IX+4),N  ;SAVE RESULT IN WKS4,5
        LD (IX+5),L
        RET
;
; SETUP WKS6
SETUPWKS6:
        LD A,(IX+3)  ;Y'
        LD B,(IX+1)  ;Y
        SUB B        ;A = Y'-Y
        INC A
        LD (IX+6),A  ;WKS6 = Y'-Y
        RET
;
; SETUP WKS7
SETUPWKS7:
        LD A,(IX+2)  ;X'
        LD B,(IX+0)  ;X
        SUB B        ;A = X'-X
        INC A
        LD (IX+7),A  ;WKS7 = X'-X
        RET
;
; SETUP WKS8
SETUPWKS8:
        LD A,WIDTH   ;SCREEN WIDTH
        LD B,(IX+7)  ;X'-X
        SUB B
        LD (IX+8),A  ;WKS8 = WIDTH - (X'-X)
        RET
;
ROWNO:
; Multiply row (Y) by number of cols (X)
; Double length shift of L by factor shift
; (10 for VDU/2K)
; Result in HL, BC corrupted
;
        LD B,SHIFT   ;SHIFT FACTOR
        LD A,00      ;USE A AS UPPER BYTE OF DOUBLE
                     ;LENGTH REG
        CCF
RDLOOP:SLA A
        SLA L
        ADC A,00
        DEC B
        JP NZ,RDLOOP
        LD H,A
        RET  :
;*************************************************
; WRITE (CHAR,STREAM,[X,Y])
; FUNCTION: This is the main control routine of
; the WRITE function. The character will be
; written at the current cursor position on the
; nominated part screen. If appropriate a CRLF will
; be done and if the last line the screen will be
; scrolled. The following control characters
; may be used in text (normal) mode
;  Clear screen  :  0CH
;  Carriage ret  :  0DH
;  Line feed     :  0AH
;  Back space    :  08H
;  Horiz tab     :  09H
;  Vert tab      :  0BH
;  Delete        :  7FH
; The following device control codes are also used
; as they are few in number rather than use escape
; sequences
;  Cursor toggle : DC1 - 11H
```

```
;  Mode toggle   : DC2 - I2D
;  Hoee cursor   : DC3 - I3H
;  Position cur  : DC4 - I4H
; The parameters [X,Y] are only used for position
; cursor
; INPUTS: A=Character to write or a control
;           character
;         B=Stream number for part screen
;         D=X coordinate
;         E=Y coordinate
; OUTPUTS: A=Error code
;*********************************************
WRITE: PUSH AF
       LD A,B          ;STREAM NO
       CP NXPT+1       ;MAX NO OF PART SCREENS
       JP M,MAXOK      ;LESS THAN NXPT
       POP AF          ;DISCARD
       LD A,STRTDBIGN
       JP EXITW
MAXOK: LD IX,WKSP      ;GET ADDR OF WORKSPACE
       LD (IX+I3),D    ;SAVE X,Y COORDS
       LD (IX+I4),E
       CALL GETSCB     ;CONVERT STREAM NUMBER TO
                       ;SCB ADDRESS
       LD A,(IY+SCBSTR)  ;CHECK FOR STREAM
                         ;ALLOCATED
       CP BFFH
       JP NZ,CONT2     ;ALLOCATED
       POP AF          ;DISCARD
       LD A,STRNOTOPEN   ;ERROR CODE
       JP EXITW
CONT2:
;
; SCB ADDRESS IN IY, WORKSPACE ADDRESS IN IX
; SET UP WORKSPACE AS FOLLOWS ...
;    WKS0 = X (COL)
;    WKS1 = Y (ROW)
;    WKS2 = X'(COL')
;    WKS3 = Y'(ROW')
;    WKS4&5= BASE + (Y*WIDTH) + X
;    WKS6 = Y'-Y
;    WKS7 = X'-X
;    WKS8 = WIDTH - (X'-X)
;
       LD B,(IY+SCBSSH)
       LD (IX+0),B
       LD B,(IY+SCBSSV)
       LD (IX+1),B
       LD B,(IY+SCBSEH)
       LD (IX+2),B
       LD B,(IY+SCBSEV)
       LD (IX+3),B
       CALL SETUPWKS45
       CALL SETUPWKS6
       CALL SETUPWKS7
       CALL SETUPWKS8
;
; NOW CHECK FOR CURSOR POSITION COORDS
       POP AF          ;GET THE CHAR
       PUSH AF
       CP I4H          ;IS IT A POSITION CURSOR?
       JP NZ,CONTI     ;NO
       LD A,(IX+7)     ;(X'-X)+1
       DEC A           ;X'-X
       LD D,(IX+I3)    ;MOVE CUR X COORD
       LD E,(IX+I4)    ;INTO Y
       CP D            ;X COORD GREATER THAN PART
                       ;SCREEN WIDTH
       JP M,ERR6
       LD A,(IX+6)     ;(Y'-Y)+1
       DEC A           ;Y'-Y
       CP E            ;Y COORD GREATER THAN SCREEN
                       ;DEPTH
       JP M,ERR7
       JP CONTI
ERR6:  POP AF          ;DISCARD
       LD A,XCURINV
       JP EXITW
ERR7:  POP AF          ;DISCARD
       LD A,YCURINV
       JP EXITW
CONTI:
;
; NOW THE REAL CODE
       CALL GETADDUNCUR  ;CONVERT CURSOR COORDS TO
                         ;ADDRESS
       POP AF            ;CHAR TO WRITE
       BIT SCBBOR,(IY+SCBATI)  ;GRAPHICS MODE?
       LD B,A
       JP NZ,WRITABLE    ;YES
       LD B,A
       CP 82H            ;IS IT A CONTROL CHAR?
       JP P,NOTCONTROL   ;NO
       LD A,B            ;RESTORE CHARACTER
       CALL CONTRDL      ;NOT WRITABLE DO DO THE
                         ;NECESSARY
       JP ENDWRITE       ;FINISHED
NOTCONTRDL:
       LD A,B
       CP 7FH          ;DELETE CHARACTER?
       JP NZ,WRITABLE  ;NO
       CALL DELETE
       JP ENDWRITE
;
; FINALLY THE BIT WHERE WE WRITE THE CHARACTER
WRITABLE:
       LD A,B
       LD H,(IX+9)     ;HIGH BYTE OF ADDRESS UNDER
                       ;CURSOR
       LD L,(IX+I0)    ;LOW BYTE
       LD (HL),A       ;WRITE THE CHAR TO SCREEN
       CALL CHKEND     ;DO THE NECESSARY FOR EOL OR
                       ;EOS
ENDWRITE:
       LD A,SUCCESS
EXITW:RET              ;ALL DONE
;*********************************************
; GETSCB (STREAM,WORKSPACE)
; FUNCTION: This routine returns the address of
; the SCB corresponding to the stream number
; INPUTS: B=Stream number
;         IX=Workspace address
; OUTPUTS: IY=Address of SCB
;*********************************************
GETSCB:LD IY,SCBL      ;ADDRES OF SCB LIST
       LD A,B
       SUB I
       LD C,A
       SLA C           ;MAKE STREAM A WORD OFFSET
       LD B,00         ;READY FOR ADD
       ADD IY,BC       ;IY=ENTRY IN SCBL
       LD L,(IY+0)     ;L=LOW BYTE OF ADDR
       LD H,(IY+I)     ;H=HIGH BYTE
       PUSH HL
       POP IY          ;RESULT RETURNED IN IY
       RET
;*********************************************
; GETADDUNCUR (WORKSPACE)
; FUNCTION: Return the address of the screen
; location under the current cursor position
; INPUTS: IX= Workspace address
;         IY= SCB address
; OUTPUTS: WKS9,I0= Address of char under cursor
;*********************************************
GETADDUNCUR:
       LD L,(IY+SCBCUY)  ;CURSOR Y COORD
       CALL RDWND        ;CALC ROW*WIDTH
       LD BC,BASE
       ADD HL,BC         ;BASE+(Y*WIDTH)
       LD C,(IY+BCBCUX)  ;CURSOR X COORD
       LD B,00
       ADD HL,BC         ;BASE + (Y*WIDTH) + X
       LD (IX+9),H       ;LOAD RESULT
       LD (IX+I0),L
       RET
;*********************************************
; CHKEND (WORKSPACE,SCB)
; FUNCTION: The character having been written to
; the part screen the current cursor position is
; incremented and checked to see if a line or
; screen boundary has been exceeded. If so the
; screen is scrolled and the cursor position
; adjusted. If the part screen was a single row
; the screen is scrolled horizontally.
; INPUTS: IX = Workspace address
;         IY = SCB address
; OUTPUTS: none
; LOGIC: IF SCBCUX=X' THEN
;          IF Y=Y' THEN
;            CALL SCURSOR
;            CALL TICKERTAPE
;          ELSE
;            SCBCUX=X
;            IF SCBCUY=Y' THEN
```

```
|              CALL BCURSOR
|              CALL 0CROLL
|            ELSE
|              INC SC0CUY
|            FI
|          FI
|        ELSE
|          INC SC0CUX
|        FI
|        CALL SCURSDR
|*****************************************
CHKEND:
|IF SC0CUX=X' THEN
       LD A,(IY+0C0CUX)   |CURSOR COL X
       LD 0,(IY+0C00EH)   |X'
       CP 0               |CUR0DR CDL EXCEEDED EDL
       JP Z,EDL           |YES
|ELSE
|   CALL SCURSOR
|FI
       IHC (IY+SC0CUX)    |HEXT CURSDR PD0ITIDN
       JP DONEI
|
|HOW PERFORM THE EOL PRDCEBSIHO AHD CHECK FDR EOS
|  IF Y'=Y THEN
|    CALL SCURSOR
|    CALL TICKERTAPE
EOL:   LD A,(IY+0C0SSV)   |Y
       LD 0,(IY+0C0SEV)   |Y'
       CP S               |I0 THIS A 0IHDLE ROW
                          |PART SCREEN?
       JP HZ,MDRETHAHDHE  |HD
       CALL TICKERTAPE    |HDRIZDHTAL SCRDLL
       JP DDHEI
|
|THRDW A CRLF AHD CHECK FOR ED0
|SC0CUX=X
MDRETHANDHE:
      LD A,(IY+SCSSSH)   |CURSOR CDL
      LD (IY+SCBCUX),A   |RE0ET
|IF SC9CUY=Y' THEH
      LD A,(IY+SC0CUY)   |CURSDR ROW
      LD 0,(IY+SC0SEV)   |Y'
      0UB 0              |0C0CUY=Y?
      JP HZ,HDTEOS       |HO,HDT EHD DF SCREEH
      CALL SCRDLL        |BCROLL PART SCREEH
      JP DDNEI
|
|IHC SC0CUY
NDTEDS:IHC (IY+9C0CUY)   |NEXT CURSDR ROW
DDHEI: CALL 0CURSDR      |CUR0OR DH IF REQUIRED
       RET               |ALL ODHE
|*****************************************
| TICKERTAPE (WDRKSPACE)
| FUHCTIOH: Horizontally acroll the screen to the
| left one character to produce a tickertape
| effect
| INPUT0: IX=Workepace poInter
| DUTPUT0: none
| LOOIC: to:=base+(Y*width)+X
|        from:=to+I
|        FDR n=ITO((X'-X)-I)
|        DO
|          [to]:=[from]
|        DD
|        [froe]:=apace
|*****************************************
TICKERTAPE:
     LD D,(IX+4)
     LD E,(IX+5)    |TD
     LD H,00
     LD L,01
     ADD HL,DE      |FRDM
     LD 0,B0
     LD C,(IX+7)    |((X'-X)+1)
     DEC 0C         |X'-X
     LDIR           |SCRDLL THE LIHE
     DEC HL         |0UMPED DHE TOO FAR
     LD (HL),02BH   |CLEAR LAST CELL
     RET
|*****************************************
| SCRDLL (WORKSPACE)
| FUNCTIDH: Scroll the part screen up one IIne, and
| clear the botton IIne
| IHPUTS: IX=Workspace pointer
| OUTPUTS: none
| LOOIC: to:=base+(Y*wIdth)+X
|        froe:=to+32
|        FOR n=ITD((Y'-Y)-i)
|        DO
|          FDR m=Ito(X'-X)
|          DD
|            [to]:=[froe]
|            Inc to
|            inc froe
|          OD
|          to:=to+width-(X'-X)
|          froe:=to+32
|        DD
|        FOR n=ITD(X'-X)
|        DD
|          [froe]:=apace
|          dec froe
|        DD
|*****************************************
SCROLL:LD D,(IX+4)     |TO=BASE+Y*WIDTH
       LD E,(IX+5)     |DE=TD ADDRES
       LD H,00         |FROM=TD+32
       LD L,32
       ADD HL,DE       |NL=FROM ADDRE0S
       LD A,(IX+6)     |FDR N=ITO((Y'-Y)-I)
       SU0 01H         |DUTER COUHT FOR MOVE

OUTER1:LD 0,00         |FDR M=ITD(X'-X)
       LD C,(IX+7)     |IHHER COUHT FOR MOVE
                       | DO
                       | [TO]:=[FRDM]
                       | IHC TD
                       | IHC FROM
                       | DD
       LDIR            |MDVE LIHE UP
       DEC A           |MDRE LIHE0 TD MOVE?
       JP I,DOHE2      |HD

       LD H,0B         |TD:=TD+WIDTH-(X'-X)
       LD L,(IX+0)     |WIDTH-(X'-X)
       ADD HL,DE       |TD+
       PUSH HL
       PDP DE          |HEW LINE ADDRES0 TD MOVE TD
|
       LD L,32         |FRDM:=TO+32
       ADD HL,DE       |HEW LINE ADDRE00 TO MOVE FROM
       JP DUTERI       |MDVE HEXT LIHE
|
|CLEAR BDTTDM LIHE REVERSE ORDER
DDHE2: LD 0,(IX+7)     |COUHT TD CLEAR X'-X
       DEC HL
CLEAR1:LD (HL),020H    |CLEAR CELL
       DEC HL
       DEC 0
       JP NZ,CLEAR1
       RET
|*****************************************
| SCURSDR (SC0 ADDRES0)
| FUNCTIDH: Reset the cursor block after an event
| which eay have resulted In the cursor moving
| IHPUTS: IY=SC0 Addreaa
| DUTPUTS: none
|*****************************************
SCURSDR:
       BIT SC00CU,(IY+SC0ATI)  ;CURSDR DH?
       JP HZ,HDTON         |HD,TURHED DFF 0Y CALLER
       CALL DETADDUNCUR    |ADDR UHDER CURSOR IH HL
       LD A,(HL)           |DET CHAR @ CUR0DR
       XOR 0BBN            |IHVERT CNAR
       LD (HL),A           |WRITE 0ACK
NOTON: RET
|*****************************************
| COHTROL (CHAR,SC0,WORKSPACE)
| FUNCTIDH: This routine jumps to the appropriate
| function according to the control code uaing
| the juap table
| CDNTTAB
| INPUTS: A=Control Character
|         IY=SC0 Addreaa
|         IX=Workapace Addreaa
| DUTPUTS: None
|*****************************************
CONTROL:LD B,A
        CP MAXCONT    |OUT OF RANGE?
        JP P,ENDCONT  |YES
        LD N,BB
```

```
        LD L,A
        SLA L           ;DONT BOTHER WITH CARRY
        LD BC,CONTROLTAB
        ADD HL,BC
        LD C,(HL)
        INC HL
        LD B,(HL)
        LD H,B
        LD L,C
        JP (HL)         ;ENTER CONTROL ROUTINE
ENDCONT:RET
;
; HERE FOLLOW THE ACTUAL CONTROL ROUTINES
;-----------------------------------------------
; CLEAR THE PART SCREEN
;-----------------------------------------------
CS:     LD B,(IY+SCBSSH)  ;X
        LD (IY+SCBCUX),B  ;RESET SCBCUX TO X
        LD B,(IY+SCBSSV)  ;Y
        LD (IY+SCBCUY),B  ;RESET SCBCUY TO Y
        CALL CLEAR
        CALL SCURSOR      ;AND RESET IT
        JP ENDCONT
;-----------------------------------------------
; HOME CURSOR
;-----------------------------------------------
HC:     CALL SCURSOR
        LD B,(IY+SCBSSH)  ;X
        LD (IY+SCBCUX),B  ;RESET SCBCUX TO X
        LD B,(IY+SCBSSV)  ;Y
        LD (IY+SCBCUY),B  ;RESET SCBCUY TO Y
        CALL SCURSOR
        JP ENDCONT
;-----------------------------------------------
; TOGOLE THE MODE BETWEEN TEXT AND GRAPHICS
;-----------------------------------------------
MT:     LD A,(IY+SCBATI)  ;FLAGS WORD
        XOR 1
        LD (IY+SCBATI),A  ;MODE BIT INVERTED
        JP ENDCONT
;-----------------------------------------------
; TOGGLE THE CURSOR BETWEEN VISIBLE AND INVISIBLE
;-----------------------------------------------
CT:     BIT 0,(IY+SCBATI)  ;IF ON THEN
        JP NZ,CUROFF       ;TURN IT OFF
        CALL SCURSOR       ;ON SCREEN
CUROFF: LD A,(IY+SCBATI)   ;FLAGS WORD
        XOR 2
        LD (IY+SCBATI),A   ;CURSOR BIT INVERTED
        CALL SCURSOR
        JP ENDCONT
;-----------------------------------------------
; POSITION CURSOR
;-----------------------------------------------
PC:     CALL SCURSOR
        LD A,(IX+13)      ;RELATIVE X COORD
        ADD A,(IY+SCBSSH) ;ABSOLUTE X COORD
        LD (IY+SCBCUX),A  ;RESET CUR X
        LD A,(IX+14)      ;SAME FOR Y
        ADD A,(IY+SCBSSV)
        LD (IY+SCBCUY),A
        CALL SCURSOR      ;TURN ON IF ENABLED AT
                          ;NEW COORDS
        JP ENDCONT
;-----------------------------------------------
; CARRIDGE RETURN
;-----------------------------------------------
CR:     CALL SCURSOR
        LD A,(IY+SCBSSH)  ;X
        LD (IY+SCBCUX),A  ;CURSOR AT START OF LINE
        CALL SCURSOR
        JP ENDCONT
;-----------------------------------------------
; LINE FEED
;-----------------------------------------------
LF:     CALL SCURSOR
        LD A,(IY+SCBCUY)  ;IF THE BOTTOM LINE
        SUB (IY+SCBSEV)   ;THEN CALL SCROLL
        JP NZ,NOTBOTTOM   ;ELSE INCREMENT SCBCUY
        LD A,(IY+SCBSSV)  ;DONT SCROLL IF
        CP (IY+SCBSEV)    ;A SINGLE LINE
        JP Z,ENDLF
        CALL SCROLL
        JP ENDLF
NOTBOTTOM:INC (IY+SCBCUY)
ENDLF:  CALL SCURSOR
        JP ENDCONT
;-----------------------------------------------
; HORIZONTAL TAB
;-----------------------------------------------
HT:     CALL SCURSOR
        CALL CHKENO  ;DOES THE CORRECT PROCESSING
        JP ENDCONT
;-----------------------------------------------
; VERTICAL TAB
;-----------------------------------------------
VT:     LD A,(IY+SCBCUY)  ;IF SCBCUY=Y THEN
        SUB (IY+SCBSSV)   ;NULL ELSE
        JP Z,ENDVT        ;DEC SCBCUY
        CALL SCURSOR      ;AND RESET THE CURSOR
        DEC (IY+SCBCUY)
        CALL SCURSOR
ENDVT:  JP   ENDCONT
;-----------------------------------------------
; BACKSPACE
;-----------------------------------------------
; IF SCBCUX=X THEN
;   IF SCBCUY=Y THEN
;     NULL
;   ELSE
;     CALL SCURSOR
;     SCBCUX=X'
;     DEC SCBCUY
;     CALL SCURSOR
;   FI
;   CALL SCURSOR
;   DEC SCBCUX
;   CALL SCURSOR
; FI
BS:     LD A,(IY+SCBCUY)
        SUB (IY+SCBSSH)
        JP Z,COLZERO
        CALL SCURSOR
        DEC (IY+SCBCUX)
        CALL SCURSOR
        JP ENDBS      ;END OF JOB FOR CURSOR NOT
                      ;LINE START
COLZERO: LD A,(IY+SCBCUY)
        SUB (IY+SCBSSV)
        JP Z,ENDBS        ;SCBCUY=Y, NO ACTION IF
                          ;CURSOR HOME
        CALL SCURSOR
        LD A,(IY+SCBSEH)
        LD (IY+SCBCUX),A  ;SCBUX=X'
        DEC (IY+SCBCUY)   ;CURSOR UP ONE
        CALL SCURSOR      ;RESET CURSOR
ENDBS:  JP ENDCONT
;-----------------------------------------------
; DELETE (DESTRUCTIVE BACKSPACE)
;-----------------------------------------------
; CLEAR CELL UNDER CURSOR
; IF SCBCUX=X THEN
;   IF SCBCUY=Y THEN
;     NULL
;   ELSE
;     SCBCUX=X'
;     DEC SCBCUY
;   FI
; ELSE
;   DEC SCBCUX
; FI
; CALL SCURSOR
;
DELETE: CALL GETADDUNCUR
        LD (HL),20H       ;CLEAR CELL
        LD A,(IY+SCBCUX)  ;CURSOR X
        SUB (IY+SCBSSH)   ;X
        JP NZ,NOTCOLZERO
        LD A,(IY+SCBCUY)  ;CURSOR Y
        SUB (IY+SCBSSV)   ;Y
        JP Z,ENDDEL       ;CURSOR HOME
        LD A,(IY+SCBCEH)  ;RESET CURSOR COORDS
        LD (IY+SCBCUX),A
        DEC (IY+SCBCUY)
        JP ENDDEL
NOTCOLZERO:DEC (IY+SCBCUX)
ENDDEL:CALL SCURSOR
        RET
;-----------------------------------------------
; CLOSE STREAM
; FUNCTION: This routine will deallocate the SCB
; but takes no other action
; INPUTS: A=Stream number
; OUTPUTS: A=Error code
```

```
;*************************************************
CLOSE:  LD A,B
        CP MXPT+1   ;SEE IF IN RANOE
        JP M,MAXPTOK
        LD A,STRTOBIOC
        JP ENOCL
MAXPTOK:CALL GETSCB ;OET ADOR OF SCB
        LD A,(IY+SCBSTR) ;CHECK IF IT WAS OPEN
        CP 0FFH
        JP Z,ERRB   ;NOT OPEN, SO CANT CLOSE
        JP CONTCLOSE
ERRB:   LD A,STRNOTOPNC
        JP ENOCL
CONTCLOSE:LD A,NOTALL      ;NOT ALLOCATED
        LD (IY+SCBSTR),A ;RELEASE SCB
        LD A,SUCCESS
ENOCL:  RET
        END
;*************************************************
```

APPLICATION 1 OF THE PART SCREEN HANDLER

```
;*************************************************
;Simple dsmonatration program for VOUK/2K Part
; scresn driver. Nine Part acreena are 'opsned
; (could be up to tsn), firat bsing the whols
; acreen. The prograe cheata (to aave my typing
; effort) in that no checking of return codea ia
; done,and it ia assumed ths streams are opened in
; sequencial order. This is only valid for a newly
; initialised prograe.
;*************************************************
        ORO  BBBBN
        LOAO BBBBH
;
;SOME EQUATES TO LINK TO EXTERNAL SUBROUTINES
OPEN:   EQU BAB90H
CLOSE:  EQU BA46AN
WRITE:  EOU BA1C0H
OETKEY: EQU  B6E2N
;
;BENERAL EOUATES
MAXST:  EOU 9       ;CAN BE UP TO 1B
;
;VARIABLES
CNSV:   OS 1
CTSV:   OS 1
;
;STREAM OATA IN THE ORDER X,Y,X',Y' FOR EACN
;STREAM
STOATA: DB B,B,31,23   ;WHOLE SCREEN
        OB B,0,0,23    ;SINOLE COLUMN IN ROW ZERO
        OB 31,B,31,23  ;SINOLE COLUMN IN ROW 31
        OB 2,0,29,0    ;SINGLE ROW IN COLUMN ZERO
        OB 2,23,29,23  ;SINGLE ROG IN COLUMN 23
        OB 5,5,8,8     ;EYE
        DB 23,5,26,8   ;OTHER EYE
        OB 14,10,17,15 ;NOSE
        OB 7,17,24,19  ;MOUTH
;
;FILL TNE PARAMETERS
FILL:   LD B,(IX+0)
        INC IX
        LO C,(IX+0)
        INC IX
        LO O,(IX+0)
        INC IX
        LO E,(IX+0)
        INC IX
        RET
;
;OPEN THE STREAM
OPENST: PUSN IX
        CALL OPEN
        POP IX
        RET
;
;OPEN ALL STREAMS
OPENALL:LD IX,STOATA
        LD A,MAXST
NEXTOPEN:CALL FILL
        PUSN AF
        CALL OPENST
        POP AF
        DEC A
        JP NZ,NEXTOPEN
;
;TURN OFF THE CURSOR
        LO B,1      ;STREAM 1
        LD A,11H    ;TURN OFF TNE CURSOR ON
                    ;STREAM B
        CALL WRITE
;
;NOW BET A CNARACTER FROM TNE KEYBOARO AND WRITE
;IT TO EACH STREAM IN TURN.
KEY:    CALL GETKEY
        JP Z,KEY    ;WAIT FOR A KEY
        LO (CHSV),A ;SAVE CHAR
        LO A,2      ;STREAM 2
        LO B,A      ;MUST BE IN B FOR CALL
        LO (CTSV),A ;SAVE CURRENT STREAM
        LO A,(CHSV) ;RESTORE CHAR
WRITENEXT:CALL WRITE
        LO A,(CTSV) ;OET THE STREAM BACK
        INC A
        CP MAXST+1  ;ALL OONE
        JP Z,KEY    ;YUP, WAIT FOR NEXT KEY
        LD (CTBV),A ;SAVE NEXT STREAM
        LO B,A      ;READY FOR WRITE
        LO A,(CHSV) ;OET THE CNARACTER BACK
        JP WRITENEXT;TNE ONLY WAY OUT OF HERE IS
                    ;TO HIT RESET!!!
        END
;*************************************************
```

APPLICATION 2 OF THE PART SCREEN HANOLER

```
;*************************************************
;A aimple electronic typewriter, partly uaeful and
; partly aa a deeonatration of a uae for the Part
; acreen handler.
;
;A  line at a time ia entered,  may be edited (no
; insert I'm  afraid),  and  ia  then  printed  by
; preaaing CR,the bottoe 22 linea keep a acrolling
; diaplay of what haa been printed so far.
;
;The  program ia exited by ETX (usually  generated
; by CNTRL & C,  thia will Cloae the  Part  acreen
; bsfore jumping to ths ZYMON ware start address.
;
;Note, if your printer requirea only CR to throw a
; new  line then you will have to rseove the LF in
; ths main loop.
;
        ORO  0B000H
        LOAO BBB0BH
;
;EOUATES FOR LINKING TO EXTERNAL ROUTINES
OPEN:   EQU BAB90H
CLOSE:  EQU BA46CN
WRITE:  EQU 0A1CDH
OETKEY: EQU 06E2H
ERR:    EQU 06A4N
WARM:   EQU BB66N
;
;BENERAL EQUATES
MAXST:  EQU 3       ;MAX BTREAMS TO OPEN
WIDTH:  EQU 64      ;WIOTN OF BCREEN
LINEB:  EQU 0F000H  ;START AOORESS OF LINE B
ETX:    EQU 3       ;ENO PRINT
;
;COORDINATES FOR PART SCREENS
SCRCOOR:OB 0,0,63,0  ;LINE B
        OB 0,2,63,23 ;LINE 2-23
        OB B,1,63,1  ;LINE 1
;
;MAIN CODE
OPEN1:  LD IX,SCRCOOR ;COORDINATE OATA
        LO A,MAXST    ;SCREENS TO OPEN
OPENLP: PUSH AF
        CALL FILL     ;FILL PARAMETERS
        CALL OPENST   ;OPEN STREAMS
        POP AF        ;COUNT
        DEC A
        JP NZ,OPENLP
WRITELN:LO B,3        ;STREAM 3
        LD O,6        ;OUTER LOOP COUNT
OUTER:  LD C,10       ;INNER LOOP COUNT,CHARS 0-9
```

```
        LD A,0B0H       ;START AT INVERTED 0
INNER:  CALL WRITECH
        INC A
        DEC C           ;END?
        JP NZ,INNER
        DEC D           ;FINISHED ALL?
        JP NZ,OUTER     ;NO
        LD A,000H
        CALL WRITECH
        INC A
        CALL WRITECH
        INC A
        CALL WRITECH
FOREVER:CALL GETKEY
        JP Z,FOREVER    ;LOOP IF NO KEY
        LD 0,A
        CP 13           ;KEY IS CR?
        JP Z,CR         ;YES
        LD A,0
        CP ETX          ;END OF SESSION?
        JP Z,FINISH     ;YES
        JP NOTCR
CR:     CALL PRNCOPY    ;COPY LINE 0 TO PRINTER
        LD B,2          ;STREAM 2
        LD A,13         ;CR
        CALL WRITECH
        LD A,0AH        ;LF
        CALL WRITECH
        JP FOREVER      ;KEEP GOING
NOTCR:  LD A,0
        LD 0,1          ;STREAM 1
        CALL WRITECH
        JP FOREVER      ;KEEP GOING
FINISH: LD 0,1          ;STREAM 1
        CALL CLOSE
        LD 0,2          ;STREAM 2
        CALL CLOSE
        LD 0,3          ;STREAM 3
        CALL CLOSE
        JP WARM         ;RE-ENTER ZYMON
;-----------------------------------------------
;SUB ROUTINES
FILL:   LD B,(IX+0)
        INC IX
        LD C,(IX+0)
        INC IX
        LD D,(IX+0)
        INC IX
        LD E,(IX+0)
        INC IX
        RET
;
OPENST: PUSH IX
        CALL OPEN       ;OPEN THE STREAM
        BIT 7,A         ;ERROR RETURNED
        JP Z,STOK       ;NO
        JP ERR
        JP WARM
STOK:   POP IX
        RET
;
WRITECH:PUSH AF
        PUSH BC
        PUSH HL
        PUSH DE
        CALL WRITE      ;WRITE CHAR TO STREAM IN B
        POP DE
        POP HL
        POP BC
        POP AF
        RET
;
PRNCOPY:LD C,WIDTH-1    ;COUNT
        LD HL,LINE0     ;LINE0 ADDRESS
PRNCPLP:LD A,(HL)       ;GET CHAR FROM SCREEN
        LD B,A
        AND 00H         ;END OF TEXT
        JP NZ,ENDPR
        LD A,0
        LD 0,A
        CALL PRINTLN    ;PRINT THE LINE
        LD A,B
        LD 0,2          ;STREAM 2
        CALL WRITECH
        DEC C
        INC HL          ;FOR NEXT CHAR
        JP NZ,PRNCPLP   ;NEXT CHAR

ENDPR:  LD A,0CH        ;CLEAR SCREEN CODE
        LD 0,1          ;STREAM 1
        CALL WRITECH
        LD A,13         ;CR
        CALL PRINTLN    ;TO PRINTER
        LD A,0AH        ;LF
        CALL PRINTLN    ;TO PRINTER
        RET
;
PRINTLN:PUSH AF
        PUSH BC
        PUSH AF
NOTRD:  IN A,(6)        ;PRINTER STATUS PORT
        BIT 7,A         ;BUSY
        JP Z,NOTRD
        POP AF
        OUT (7),A
        POP BC
        POP AF
        RET
;
        END
;********** THE END ***************************
```

(Ed- See the letter from Bob in the letters section for some extra notes about this Part screen display project.)

AUNTIE DAVIDS PAGE

BETRAYED!

If I find that guy he's for it. Let me know if you've seen him. He crept into our house some time late last year, a big fat fellow with a red cloak and a white cotton-wool beard (I'm pretty sure it was one of his reindeer loosened one of my chimney pots) and what did he leave my elder boy Andrew but a Sinclair Spectrum 128K Plus! The shame of it.

To be fair Andrew (age 8) and his brother Robin (age 5) have had a lot of fun with it, so if I see Santa I'll let him off with a reprimand, but it is a very difficult machine to get to grips with in a serious way. I'm not sure Amstrad (who helped Santa make it) know what they're putting in the box; half the tapes supplied don't load first time, or "crash" just in the middle of the interesting bit. As part of their policy of continuous improvement Amstrad have rewired the joystick ports (so you can't use any joystick you already might have); the only trouble is many of the games in the kit were written before the "improvement" was made so will work with any joystick but the one supplied!

Inside is a miracle of consumerism. (I had to look inside on Christmas day to try and see if there was any reason for the unexplained crashes.) On a board hardly any bigger than a single Interak pcb is the whole computer. I recognised the sound chip (AY-3-8912) the modulator (UM 1233) memories (4164) a few LS parts and the CPU (Z80A), but the rest were all unknown "Amstrad" specials, some with dirty great heatsinks grafted on as if an afterthought. There is little use of sockets, no circuit diagram, and no source of replacement chips, so I had to give up and screw it back together.

It later turned out that it was mostly defective software causing the trouble. Andrew's school friend (whom Santa had similarly favoured) had the same sort of trouble but on different games in the pack. Evidently the tape loading (even with the built in recorder) is unreliable, and little or no error checking is carried out, so you can easily get a bad load without knowing about it until the game is well under way. The simple solution of making fresh copies of the games which do work is not possible because the tapes are copy protected - if you have an accident or a defect develops in the original that's your lot!

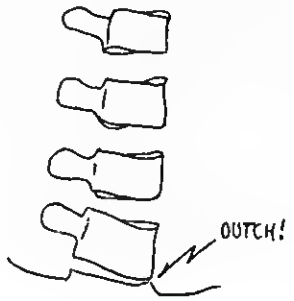
Why am I confessing to having an Amstrad Sinclair in the house, when I really should have nothing but Interaks? I am like all the other Interak users with a fierce loyalty to Interak. In fact there is no shame in having a computer other than Interak. Different computers do different jobs. The Sinclair is wonderful for example for plugging into the family TV and playing games swapped in the school yard (even though they shouldn't be, but the children see it the same as swapping comics, cricket bats and the like). An IBM is wonderful for running grown up packages in a similar way, the same with the better Amstrads, Ataris, Macs, etc. If you just want to plug in and go, there's no beating them, there never has been, so don't feel ashamed if you own a computer other than Interak, you can come out of the closet - your secrets are safe with me. Anyway from my own personal experience with Amstrads, IBMs and the like, it is always a welcome relief to get back to Interak.

MIND YOUR BACKS

I am now getting a few letters addressed to "Auntie" at Greenbank. (A fond hello to all my new nephews, no nieces yet!) Circumstances have forced me to take my partial sex-change a bit too far; as I sit and write this I am wearing my new corset. This is not what you think you bad minded jokers, the new underwear is entirely medical. I have been suffering with a bad back and some sciatic pain in my right leg since last July.

The diagnosis is an arthritic spine caused by wear and tear. Apparently many years ago I suffered a "slipped" disk without realising it (this is possible because there are no pain receptors in the disks - you only feel it if a nerve is trapped), and it has gradually deteriorated unknown to me until last summer I pushed it over the brink by using a starting-handle on my car when it had a flat battery. Never again!

I took a sneak look at my X-Rays, and even as a layman the trouble is obvious. This is a sketch of what I saw:



The specialist says there is really no chance of a cure without an operation to fuse the vertebra to the bone beneath and stop any further movement.

I must give my thanks to all of you who have expressed sympathy and offered practical suggestions from their own experience. Bob Eldridge (a fellow sufferer following an accident at sea - it's amazing what people have done) suggests jogging and swimming to replace my steel corset with natural muscle support, the osteopath I have tried and my doctor however both advise against exercise whilst there is still pain (but in the last six months or more they haven't managed to cure me). George Coombs suggests a Selenium and vitamin dietary supplement which he has found helpful for his own arthritis, and has given me advice on selecting a qualified osteopath. John Meaker, a lecturer at North Humberside tells me of an almost crippled lady at a Yoga class who was rejuvenated by Yoga exercises; Charlie Bridgstock recommends pain killers and "Tiger Balm" (a remedy he has picked up in his travels in the world - ask him about these, he has some interesting tales to tell). Kevin Daley says that as a designer of Interak I should simply fit a new "backboard", and the rest of you have all offered your sympathy and support.

I can certainly say that if there was any chance that standing with one leg in a bucket of cold porridge in the middle of an elm wood at midnight facing due north and singing "We Shall Overcome" would do the trick I should be the first to try it. (It doesn't work by the way.)

Bye for now. Well done Bob on the new layout; thanks to Tom for looking after everyone; thanks to Charlie for successful launch of disk library.

SOME ZYBASIC PROGRAMS

By Paul M. Nicklin
For Zybasic

This suit contains :-
BIORHYTHMS, CALENDAR PRINTER,
SIMULATING 3D ARRAYS
SIMULATING "CODE"

Note:- [CTRL_X] means press Control and X together
[CTRL_U] means press Control and U together
[10*CTRL_Y] means press control & Y ten times.

BIORHYTHM PROGRAM

Notes :-
Line 110 inputs date of birth.
120 Calls sub at 200.
140 Input todays date.
170 Calls sub at 300.

Sub 200 calculates days since 1st Jan 0000
310 Prints graph.
340-400 Vitality graph.
The use of the point command ensures that titles etc. are not overwritten.
The pattern of the cycle is a sine wave.
This prograe would have been shorter if CODE and MODULO were available. See later in this article for simulations of these.

```
100 CLS: LINE 1: PAGE: DOFF
110 INPUT "Date of birth:","Day "D,"Month "M,
    "Year "Y
120 GOS.200
130 N1=N
140 INPUT "Today's date:","Day "D,"Month "M,
    "Year "Y
150 GOS.200
160 N=N-N1
170 GOS.310
180 STOP
200 N=Y*365
210 N=N+INT(Y/4)+INT(Y/400)-INT(Y/100)
220 C1=0: IF Y/4=INT(Y/4) C1=1
230 C2=0: IF Y/100=INT(Y/100) C2=1
240 C3=0: IF Y/400=INT(Y/400) C3=1
250 C4=(M+3)
260 C5=0: IF (C2+C3) C5=1
270 N=N-(C1*C4*C5)
280 A$="231223345566"
290 N=N+PEEK(#80FF+M)+30*(M-2)+D
300 RET.
310 N=N-32
320 GOS.640
330 GOS.600
340 LINE 1: PRINT "[CTRL_X] Line represents
    average.": PRINT "[CTRL_X] Intellect."
350 E1=E: V1=V: I1=I
360 FOR A=1 TO 128
370 Y=23+23*SIN((A/2+I1)*180/33)
380 IF POINT(A/2,Y)<>0 0.480
390 SET(A/2,Y)
400 NEXT A
410 GOS.670
420 LINE 1: PRINT "[CTRL_X] Line represents
    average.": PRINT "[CTRL_X] Emotion.  "
430 FOR A=1 TO 192
440 Y=23+23*SIN((A/3+E1)*180/33)
450 IF POINT(A/3,Y)<>0 0.470
460 SET(A/3,Y)
470 NEXT A
480 GOS.670
490 LINE 1: PRINT "[CTRL_X] Line represents
    average.": PRINT "[CTRL_X] Vitality. "
500 FOR A=1 TO 192
510 Y=23+23*SIN((V1+A/3)*180/23)
520 IF POINT(A/3,Y)<>0 0.540
530 SET(A/3,Y)
540 NEXT A
550 LINE 1: STOP
600 I=N-33*INT(N/33)
610 E=N-28*INT(N/28)
620 V=N-23*INT(N/23)
630 RET.
640 FOR A=1 TO 23: LINE A: ?"[CTRL_X]": NEXT A
650 LINE 12: ?"[CTRL_U],[15*CTRL_Y],[CTRL_W],
    [13*CTRL_Y],[CTRL_R ":
    ?"16 Days ago   Today  In 16 days"
660 RET.
670 LINE 1: ?Press a key.        "
680 INK.K
690 INK.Y: IF Y=0 0.690
700 CLS
710 GOS.640
720 RET.
```

CALANDAR PRINTER

Notes:-
Ds is sliced at line 340, Line 90 puts Sunday in inverse video. Ms contains the offset (MOD 7) that occure from assuming each month has 28 days. Ie after 3 months you would be 6 days out, so Ma(4) is 6. Ls contains the number of days greater than 28 of each month. Ls is then PEEKed to find L which will then contain the number of days in that month.

Lines 270-290 decide conditions C1-C3 and 300-acta on these conditions to produce an offset value of the number of leap days to add if the month is greater than February then these days are added else condition 4 is zeroed.

The subroutine at 70-100 READs the date and finds from them your month and puts the number of the month into e1.

N.B. Graphics instructions are shown boxed [CTRL_]

```
10 CLS: PAGE: LINE 1: DOFF
20 C=0: C1=0: C2=0: C3=0: C4=0
30 L$="303232332323"
40 INPUT "Year "Y
50 ? "Month in capitals ":W$=INPUT$
60 W$=MID$(N$,1,3)
70 M$="033614625035"
80 D$="SunMonTueWedThuFriSat"
90 PO.#8400,#D3,#F3,#EE
100 N=Y
110 RESTORE
120 M=N+INT(N/4)-INT(N/100)+INT(N/400)
130 M1=1
140 READ N$
150 IF N$=W$ 0.190
160 M1=M1+1
170 IF M1<=12 G.140
180 0.50
190 PRINT
200 LINE 6
210 C=PEEK(#8C00-I+M1)
220 IF (M1=2)*(Y/4=INT(Y/4)) C=C+1
230 FOR A=1 TO 6
240 C=28*A+1
250 P2=PEEK(#8D00+M1-1)-48
260 C=M+A+P2
270 IF N/4=INT(N/4) C1=1
280 IF N/100=INT(N/100) C2=1
290 IF N/400=INT(N/400) C3=1
300 C4=C1-C2+C3
310 IF M1<3 C4=0
320 C=C-C4
330 C=C-7*INT(C/7)
340 ?MID$(D$,3*C+1,C)
350 FOR B=0 TO 4
360 C=7*B+A+1: IF C<=L P.%3,C,
370 NEXT B
380 P.
390 NEXT A
400 LINE 2
410 P."[CTRL_S],[9*CTRL_Y],[CTRL_R] ":
    P."[CTRL_X"+N$,%5,Y,"[CTRL_X":
    P."[CTRL_Q],[9*CTRL_Y],CTRL_P "
420 LINE 16
430 STOP
500 DATA "JAN","FEB","MAR","APR","MAY","JUN",
    "JUL","AUG","SEP","OCT","NOV","DEC"
```

SIMULATION OF 'CODE'
Notes :-
X returns the code of MID$(String,P,1)
S contains code of string is. 41H for A$.

```
3700 ! CODE( $ )=X S contains code of string is
     41H for A$
3710 S=S*64: S=S*256
3720 X=PEEK(S-1+P)
3730 NEXT A
```

SIMULATION OF MODULO
```
3000 ! Y=X MOD Z
3010 Y=Z-Z*INT(X/Z)
3020 RET.
```

SIMULATING 3D ARRAYS
In most versions of Basic a provision le made for multi-dimensioned arrays. In Zybasic only one single dimensioned array exists, it is accessed by

@(number)

In IUBN 2 Pete gave us a formula for 2D arrays, 3D arrays are simply an extension of this :-

Suppose we wish to store Temperature readings for 2 weather stations over one week.

This would require a 2D array (2,7) and a given temperature would be accessed by (W,D) for day D and Station W.

To do this in Zybasic you would use the forsula :-

@((W-1)*7+D-1)

Suppose now we wish to store Rainfall totals as well, another dimension is needed so our array becomes (2,2,7) and a weather station would be accessed by (F,S,D) for station S on day D, F would be 1 for temperature, or 2 for rainfall.

In Zybasic this would be :-

@((F-1)*14+(S-1)*7+D-1)

So the general formula for an array (x,y,z) is :-

@((X-1)*(y*z)+(Y-1)*z+Z-1

The parts underlined should be replaced by the correct values.

Here is a subroutine that will put N into store (X,Y,Z). (Again the correct values should be placed in the underlined parts.)

```
1500 Z9=((X-1)*___+(Y-1)*___+Z-1
1510 @(Z9)=N
1520 RET.
```

4D ARRAYS
This formula will put N into store (W,X,Y,Z)
```
1550 Z9=((W-1)*x*y*z+(X-1)*y*z+(Y-1)*z+Z-1)
1560 @(Z9)=N
1570 RET.
```

The array size is (W,X,Y,Z) and these should be substituted into the formula.

********** THE END **********

CDMS-1 DESIGN AND CIRCUIT DESCRIPTION
By David Parkins

Introduction
This is an experimental design for a general purpose serial interface, but designed with use for data communications via modems very much in mind. It is based on the now aged "Kemitron" 810-4 card, but has some more modern additions. In particular, there are a set of individually selectable baud rates for both transmit and receive, and extra outputs and inputs to allow the use of MODEMS. Except for the selection of the block of 4 I/O Port addresses which the design uses, all selections are to be carried out by software. Because of this (and also because this card uses 4 I/O Port Addresses to control just one UART, compared with the previous 2 I/O Port Addressing per UART of the less sophisticated design) this card requires its own new software to drive it.

An earlier design "XSER2", was developed in September 1985, with the help of Tom Evans (SYSOP of the Tascom-Interak Bulletin Board) and this design is a very close development of the XSER2.

Our thanks are due to Tom Evans for his help and suggestions, and his experimental work and debugging "on line".

The main differences from the XSER2 are:

1. Provision of 16 Jumper Links to allow configuration as DTE (Data Terminal Equipment) or DCE (Data Communication Equipment).

2. Removal of option for TTL level outputs as alternatives to RS-232 levels.

3. Rearrangement of Modem handshaking port bit allocations (in order to ease printed circuit board design rather than for any other reason).

Warning
Note that this is still only an experimental design! If a printed circuit is produced it may be different in many ways from what is discussed here!

DTE and DCE Configuration
Serial communication via the 25-way D-type plug of a conventional RS-232 data link of course involves both "transmission" and "reception". Ideally one party transmits on pin 2, (the "TXD" signal line) and the other receives on the same line. Then (or simultaneously in the case of "full duplex" communication) the other party transmits on his pin 3 (the "RXD" signal line) and the first party receives on pin 3.

You can perhaps detect an anomaly here: if we transmit on the TXD line (pin 2) and receive on the RXD line (pin 3) the remote party has to receive on the TXD line and transmit on the RXD line. This is probably quite understandable to you because of course when one party transmits the other has to receive, and vice versa.

There are generally two kinds of equipment possible; "DTE" or Data Terminal Equipment, (for example a computer and its operator), and "DCE" or Data Communications Equipment, (for example a modem). By the way, I am aware that the proper meaning of DCE is "Data Circuit Terminating Equipment", so don't write in and tell me, but "Data Communications Equipment" seems more explicit to me, so that's what I shall talk it to man.

DTEs transmit on pin 2 and receive on pin 3; DCEs receive on pin 2 and transmit on pin 3. Viewed this way it makes pretty good sense. Naturally a modem which has received some data from some

distant data provider will have to transmit it to the data terminal equipment, which in turn receives it. The modem obviously will have to transmit along the data terminal's receive line if the terminal is to receive it. And I suppose, to complete the story, the message will be presented on the terminal's screen or printed on its printer for transmission optically to the user's eyeballs which receive the image to transmit to his brain cells, which receive the information.

I have laboured the point this much so that you can agree how simple it all is. There can be no such thing as an absolute definition of transmission or reception of data; what is transmission from one party ie reception by the other. The names given to the RS-232 signal lines "TXD" and "RXD" are defined according to the point of view of the DTE. DCEs obligingly fall in with the definition which suits the DTE, thus a DCE has to receive on the TXD line and transmit on the RXD line.

Before the days of the brash new computer designers a very sensible plan, which was common then, was to use a male connector to signify a DTE and a female connector to signify a DCE. Then logically simple cables (connecting pin 1 to pin 1, pin 2 to pin 2, pin 3 to pin 3, etc) male on one end and female on the other served all purposes, with no danger that the two parties would both try to transmit on the TXD line and both try to receive on the RXD line and thus fail miserably to communicate.

Life unfortunately never remains so simple. The first problem is what to call a device like a Teletype. (For those who are too young to have seen a Teletype, this is what computer operators used as a terminal in olden days. It is very like a telex machine; the operator types at a keyboard to transmit instructions and messages and the received answers are printed out on a roll of paper for the operator to read.) So what is a Teletype to be, DTE or DCE? You can hardly blame those who described it as a data terminal for that is exactly what it was. It was sensible to connect this DTE to a modem, which is definitely a DCE (nobody could dispute that!)

Later there were more sophisticated replacements for Teletypes. For example Diablo and Qume made terminals with high quality daisy wheel printers, and naturally they were DTEs as before. However a new use was simply as printout units for small computers. In this application the keyboard on the printer mechanism was redundant and more often than not printers were supplied without any keyboard at all. The problem now was that as a result of their evolution (as just explained in my history lecture) true data terminals, ie DTEs, ordinary printers inevitably had to be thought of as DTEs too. So a modern Epson dot matrix printer with a serial interface is supplied wired up as a DTE (although with a fleeting trace of unease at their ambivalent position, the 25-way D type connector has changed sex. A DTE should really be male, but as a printer can hardly now be thought of as this it has adopted a limp wristed stance and displays the female connector normally associated with a DCE.)

There is no doubt that a computer is best thought of as a DTE, but faced with the task of interfacing both to modems (definite DCEs) and printers (DTEs themselves, by tradition) many computers have a crisis of identity and are provided by their designers with DTE configured serial interfaces, but often with the female connector of a DCE. To my mind there should be two classes of serial interface on a computer, and of course this is possible with Interak which can have plugged in as many serial interfaces as will fit in a rack. The two classes of serial interface would be DTE with a male plug (for connection for example to modems), and DCE with a female connector (for connection for example to printers). Perhaps we could even allow a DTE configuration with a female connector to accommodate the limp wristed fraternity.

You will appreciate that what is an output on a DTE is an input on a DCE, so a rational design like ours will have to allow for a complete change of direction according to whether or not the configuration is to be DTE or DCE, so this has been allowed for in the CDMS-1 design.

Incidentally you may be wondering how lesser computers manage the impossible trick of connecting themselves (DTEs) to printers (also DTEs), if neither will condescend to be a DCE for the purposes of communication. The answer, at least as far as TXD and RXD are concerned, is to cross the TXD and RXD wires in the interface cable so that both may now happily transmit on their TXD output and receive on their RXD input. Although this is temptingly simple there is more to it than that. TXD and RXD are not the only signals to be considered (see the next topic for more on this) and so wholesale intertwangling of wires becomes necessary. Round shouldered professionals in data communications carry as the tools of their trade ten or twenty cables made up to the most common lunatic permutations, and have as their secret weapon a "breakout box" which is capable of connecting anything and everything to everything and anything. You will have heard of the forthcoming Stock Exchange "Big Bang"; this will be the literal consequence of letting too many data communications chaps loose with their breakout boxes.

RTS, CTS and other Handshaking Signals

Mercifully not all the 25 way signals are in common use. This is lucky because there are 30 or more definitions for the 25 signals, (plus a folklore of traditional wrong connections, where perhaps a famous computer was manufactured to a blueprint which had a coffee stain on it, and the designer didn't like to admit he'd dropped a clod!)

So far I have only discussed the transmit and receive signals, TXD and RXD, but, especially with printers and modems, you can't just go transmitting and receiving any old time. You can't send data to a printer at a rate of 1000 characters per second if it can only print at 150 characters per second; you can't go transmitting the story of your life to Bulletin Boards Anonymous if your modem is switched off, and so on.

To my simple mind much of this extra complication could be solved by software handshaking methods, for example XON/XOFF (Transmit on/Transmit off) signalling or ETX/ACK (End of Text/Acknowledge) signalling instead of hardware handshaking lines (eg the "Busy" signal from a printer to prevent the receipt of further data when its internal buffer is full). However there is no point in trying to change the world, so the CDMS-1 design is arranged to accommodate the most common hardware handshaking methods. Despite my outburst above that all handshaking can be carried out by software, this is manifestly not the case in the real world (as proved by the difficulties the "reduced" RS-232 4 or 5 wire serial interfaces found on home and small business computers get into when faced with real data communication problems).

The major two signals (in the sensible circumstance of a DTE connected to a DCE) are :-

RTS (Request to Send) and
CTS (Clear to send):

RTS (Request to Send)	Pin 4	(Output from DTE)
CTS (Clear to Send)	Pin 5	(Input to DTE)

The DTE ieauea an RTS when it has some data to send, but restrains itaelf froe blurting out the data until it receives a CTS from the DCE. When for some reason (perhaps ita internai buffer is full) the DCE cannot accept further data it denies the CTS signai to the DTE.

However, before all this can take piace there is often a more gross ievel of handshaking which has to be set up. Thia involves two more signals, DTR and DSR (again in the feiicitous connection of a DTE to a DCE):

DTR (Data Terminal Ready) Pin 20 (Dutput from DTE)
DSR (Data-Set Ready) Pin 6 (Input to DTE)

Of course somewhere in the world there will be a legal document defining these signals in tortuous detail, but just so that you can get a feei for what purpose DTR and DSR serve, you can think of the DTE as being your computer, and the DCE as your modem. Firat the DTE activates DTR to indicate to the DCE (the modee in thia exaeple) that the DCE should wake up (if it was asleep) and eake itself ready for action. When it is ready the DCE then sends the DSR signal which is an input to the DTE, indicating that there is indeed a DCE present, ready for uae. A general result of removing the DTR signal from a eodem in use is to cause the modem to disconnect from the telephone line (ie to "heng up") and effectively to go back to sieep.

Aithough there are eany other signals uaed by perpetrators of the RS-232 25 way D Type connector criee, the signals mentioned eo fer ere the eain ones in general use for hardware handshaking.

For the simple caae of a DTE (eg computer) connected to a DCE (eg Modee), the full procedure goes like this:

1. First the DTE activates DTR (Data Terminal Ready) to cali the attention of the eodem to the fact thatits services are required.

2. Next the DCE (eodem) activates DSR (Deta-Set Ready) to iet the DTE know that the DCE is ready for use.

3. The DTE (computer) issues RTS (Request to Send) to ask the DCE'a (modem's) pereission to eend data for transmiaaion down the line. (It is probably best first if the DTE checks for the presence of a vaiid data carrier for dete transmission before entering this stage. The eodem ahould provide this signal aa DCD (Data Carrier Detect); i shall go into thia in a little eore detaii iater.)

4. The DCE grants thia peraiasion (if it is abie) by activating CTS (Ciear to Send).

5. Now the DTE sends the data. (Inaide the computer itself the controiling software has to carry out a further piece of internal handahaking, checking a signal caiied TBMT (Tranamit Suffer Eapty) on the UART (the "Universai Asynchronous Receiver Tranamitter" chip), before aending data even to the UART. When inputting data the signal to be checked ia DAV (data avaiiable).)

As 1 eentioned, the above DTE to DCE connection ia the aimple (!) arrangement. ita great benefit la that a atraight 1 to 1, 2 to 2, 3 to 3, 4 to 4, etc cable connects the two equipments.

1 have already eentioned however that to confuae the iasue conaiderably DTEa are sometimea connected to apparatua which aiao wanta to be thought of aa a DTE. Thia occura eoat coemonly when a computer (as DTE) ia connected to a printer (also a DTE).

A straight connection between two unmodified DTEs is courting diaaater, eg connecting the RTS output pin 4 of one DTE to the RTS output pin 4 of the other DTE is bound not to work. (It may not result in a catastophe if, as in this deaign, appropriate arrangeeents are aade to prevent daeage.)

You will remember that the crude way to get over the "DTE connected to another DTE" problee as far as TXD (Tranamit Data) Pin 2 and RXD (Receive Data) Pin 3 are concerned is to cross over the connections, so that pin 2 of one DTE is connected to Pin 3 of the other, and vice versa.

But the plot thickens when we come to the handahaking lines. All that can be done is to take a deep breath and cross over soee more lines on a trial and error basis. An arrangeeent which sometimes works is to eake up a speciai cable, which inciudes the following (there are additionaiiy other atraight through connections which must be made, and yet other straight connections which euat not be made, but I have left them out for "ciarity" (!) and because even Solomon could not resolve soee disputes between two DTEs which both want to be the boss).

```
 2 TXD   -------->  3 RXD
 3 RXD   <--------  2 TXD
 4 RTS   -------->  5 CTS
 5 CTS   <--------  4 RTS
 6 DSR   <--------  20 DTR
20 DTR   -------->  6 DSR
```

If you think the whole idee of heving soee signals croased and eoee signals straight and some signals not connected, on pain of death, ie stupid, you are right, but there it is. In the CDMS-i design I have tried to take obvious pairs of signals as pairs, and I have organized it so that in consequence the action of converting a CDMS-i board froe DTE configuration to DCE configuration also has the effect of crossing over the signals which sometimes have to be crossed over.

Said again in other words, the CDMS-i board can either be configured as a DTE, or by rejumpering it (by means of push on shorting links) it can be reconfigured as a DCE. So far ao good. The bonus of the juepering method I have uaed ia that if the board is rejuepered es DCE, but soee Wally atiii treats it as DTE without any crossed wiree in his interface cable, my COMS-i board may stiii be ebie to cope. This is because my DCE configuration has been cunningly chosen to be the same as a DTE but with the essentiai reversals of TXD/RXD RTS/CTS DTR/DSR already eade. Thus there is often no need for a speciai crossover cable to connect a DTE or a DCE to the CDMS-i board. (This ieaves soee signals out in the cold, because they will be outputs, aay, at opposite ends of the saee wire, with no obvious coepanion input signal with which to cross over. There'a not much I can do about this, aave suggest that you leave one end diaconnected, easy to do just by leaving out the affected JLink.)

As I have said before, when this eethod of serial connection was firat deviaed it was perfectly sound, it is only the stupidity or ignorance of eanufacturers of computers and peripheral equipment (perhaps not recogniaing the diatinction between DTE and DCE, and the logic of male connectors for DTE and female for DCE) which haa reaulted in the meas we aee today. There are perhaps hundreds of different waya of connecting thinga auch as printers to computera via an RS-232 interface, and the coepiexity ia due to ioopy manufacturera uaing aome signal of their own, when there waa a perfectly good one already defined for the purpoae.

I hope you will forgive the fact that the CDMS-i deaign waa intended only to cope with the aaner

connections directly, and has tended to ignore the madder ones. (These latter can of course be patched in to individual choice by pending applicants for admission to the funny farm.)

Other Signals

I shall now mention the other signals of the 25-way D type connector used in the COMS-1 design. This leaves several which will not be mentioned, and are not built into the basic design. The reason for omitting thee ie that their presence or absence on any specific equipment is only a matter of luck (good or bad!) and if you are obliged to use thes you will have to play things by ear. You won't get much help here from me anyway.

Pin 1, Protection Ground

Pin 1 is the Protection Ground, once used for earthing a DCE connected to a DTE, but nowadays (thanks to the confusion between which of two given equipments is the DCE and which is the DTE) of less certain purpose. Anyway, in this design it is connected to 0V. it is probably safest to connect pin 1 of one equipment to pin 1 of the other through the interfacing cable.

Pin 7, Signal Ground

Pin 7 is the common, signal, ground, and should always be connected to Pin 7 of the other equipment, via the interface cable.

Pin 8, DCD

Pin 8 is DCD (Data Carrier Detect) which has so far only been mentioned briefly. it is an input to a DTE and an output from a DCE. Its name only makes any sense if the DCE is a modem, which is connected to a telephone line and receiving a valid "data carrier" tone.

Pin 11, STF

Pin 11 is STF (Select Transmit Frequencies). STF is an output from a DTE and an input to a DCE. Again it is only relevant if the DCE is a modem. Activating the STF line instructs the modem to consider itself as the originator of a telephone call (the called party is then to be treated as the answerer). In many circumstances different frequencies from those used for an originator are used for an answerer, and this allows "full duplex" operation (where simultaneous two way communication, sometimes heard in ordinary telephone speech, is allowed,) if each party has to take turns to transmit whilst the other receives this is known as "half duplex", but this subject is contained in a whole new can of worms which I don't want to open at the moment!
(Many modern modems (such as our recommended Miracle Technology WS3000 or WS4000 series) do not use the pin 11 signal, but make other arrangements via "intelligent" software commands.)

Pin 22, RI

Pin 22 is RI (Ring Indicator). This is an input to a DTE and an output from a DCE. RI only has meaning if the DCE is a modem. it is active when the modem detects the presence of a remote party ringing in. It is the logical equivalent of the ringing sound you hear from your own telephone when you have just stepped into the shower.

Pin 23, DSR

Pin 23 is DRS (Data Signalling Rate). This is an input to a DTE, and is an output from a DCE. It is another signal which is only relevant if the DCE is a modem. Not all modems provide this signal (our favourites the Miracle WS3000 and WS4000 do). The signal is produced by modems which are capable of working at more than one data rate (for example 300 baud and 1200 baud) and able to detect automatically when their highest rate is to be used when "auto answering" a call. The DRS signal from the modem is of great use to the DTE (ie to the computer in this context) because it allows the computer to switch its own baud rate to suit that of the calling party. in consequence an automatic "Bulletin Board" can be set up to receive calls at any either 300 baud or 1200 baud to suit the equipment of the calling party.

Pins 15 and 17, External Clocks

Pin 15 and 17 carry signals known respectively as Transmit Clock and Receive Clock. They are both inputs to DTE and outputs from DCE. They are additional signals which are produced by some high speed modems (say 2400 baud or more) as part of their progressively more complex modulation and demodulation arrangements. (1200 baud is near the natural limit on the ordinary telephone network, 2400 baud and above demands extra complexity like this, and usually rapidly escalating expense.)

Signal Levels and Definitions

I have (deliberately) been vague so far about the signal levels in the above descriptions. I have described signals as being "active", and being input and output to DTEs and DCEs, without mentioning their actual levels. This is because I wanted to get the meaning across in English Word-type Words, rather than Boolean Logic symbolism and other mathematical gumbo-jumbo. The particular problem in describing RS-232 level signals is that "negative" logic is often used, where a logic "0" is represented by a voltage level of around +12 volts, and a logic "1" by +12 volts. Worse still, the signals are "active low" ie represented by logic "0", ie +12 volts, ie "actlvs high" (see, i've now confused myself as well as you!) Here is a vade-mecum (ah, the benefits of a classical education) to which you can cling:

RS-232 Levels:

+12V = "ON", = "logic 0", = "Space"
-12V = "OFF", = "logic 1", = "Mark"

(The +12V and -12V voltages are "nominal" and a wide variation in these levels is tolerated (for example some home and small business computers keep costs down by using voltage levels of +5V and -5V instead of +12V and -12V)).

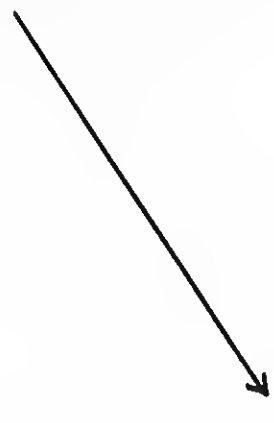
Alternative Logic Definitions:

As many of the bright young things who design modems today weren't even born when negative logic was invented, they prefer (and so do i really) to call the high (+12V) voltage level a logical "True" or "1", and the low (-12V) voltage level a logical "False" or "0":

+12V = "ON", = "logic 1", = "True" = "Space"
-12V = "OFF", = "logic 0", = "False" = "Mark"

They'll be telling me next current flows into the + terminal of a battery (well doesn't it?)

[ED- The circuit diagram for the CDM 1 card follows. A description of the circuit is after the 6 figs].



COM 1 CIRCUIT DIAGRAM (Figs 1 and 2)

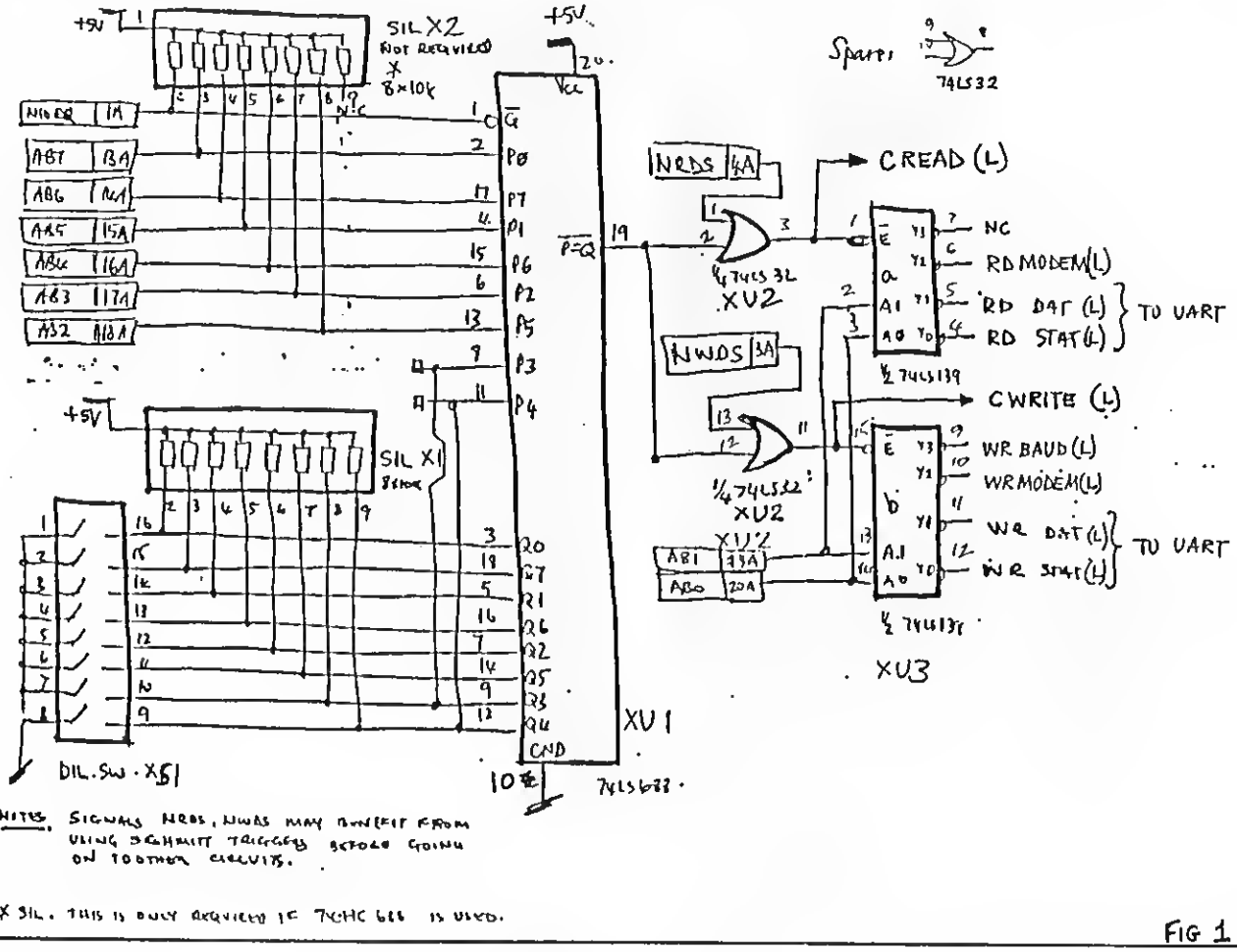
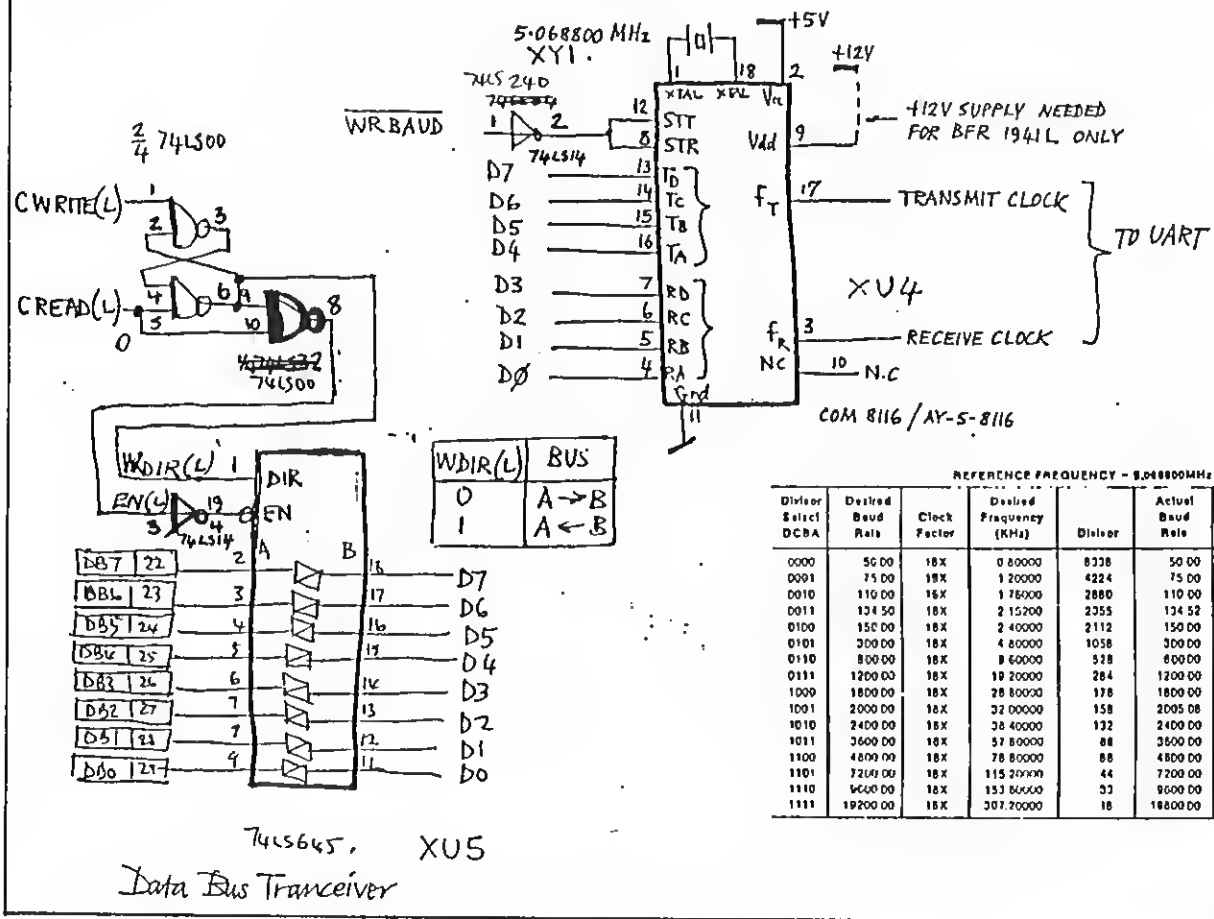


FIG 1



REFERENCE FREQUENCY = 5.068800MHz

Divisor Select DCBA	Desired Baud Rate	Clock Factor	Desired Frequency (KHz)	Divisor	Actual Baud Rate	Actual Frequency (KHz)	Deviation
0000	50.00	16X	0.80000	8338	50.00	0.800000	0.0000%
0001	75.00	16X	1.20000	4224	75.00	1.200000	0.0000%
0010	110.00	16X	1.76000	2880	110.00	1.760000	0.0000%
0011	134.50	16X	2.15200	2355	134.52	2.152357	0.0166%
0100	150.00	16X	2.40000	2112	150.00	2.400000	0.0000%
0101	300.00	16X	4.80000	1058	300.00	4.800000	0.0000%
0110	600.00	16X	9.60000	528	600.00	9.600000	0.0000%
0111	1200.00	16X	19.20000	264	1200.00	19.200000	0.0000%
1000	1800.00	16X	28.80000	176	1800.00	28.800000	0.0000%
1001	2000.00	16X	32.00000	158	2005.08	32.081013	0.2532%
1010	2400.00	16X	38.40000	132	2400.00	38.400000	0.0000%
1011	3600.00	16X	57.60000	88	3600.00	57.600000	0.0000%
1100	4800.00	16X	76.80000	66	4800.00	76.800000	0.0000%
1101	7200.00	16X	115.20000	44	7200.00	115.200000	0.0000%
1110	9600.00	16X	153.60000	33	9600.00	153.600000	0.0000%
1111	19200.00	16X	307.20000	16	19800.00	318.800000	3.1250%

FIG 2

COM 1 CIRCUIT DIAGRAM (Figs 3 and 4)

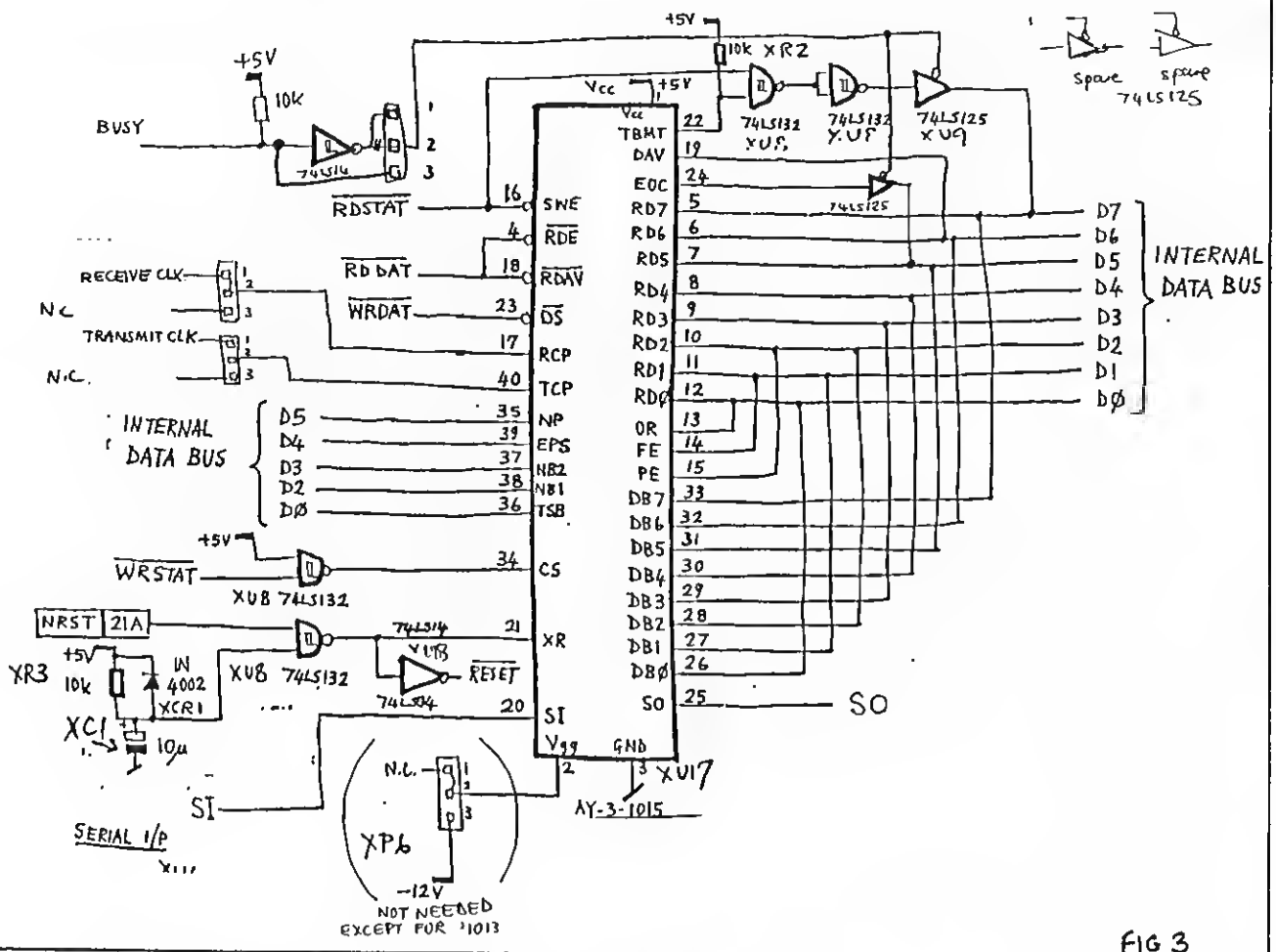


FIG 3

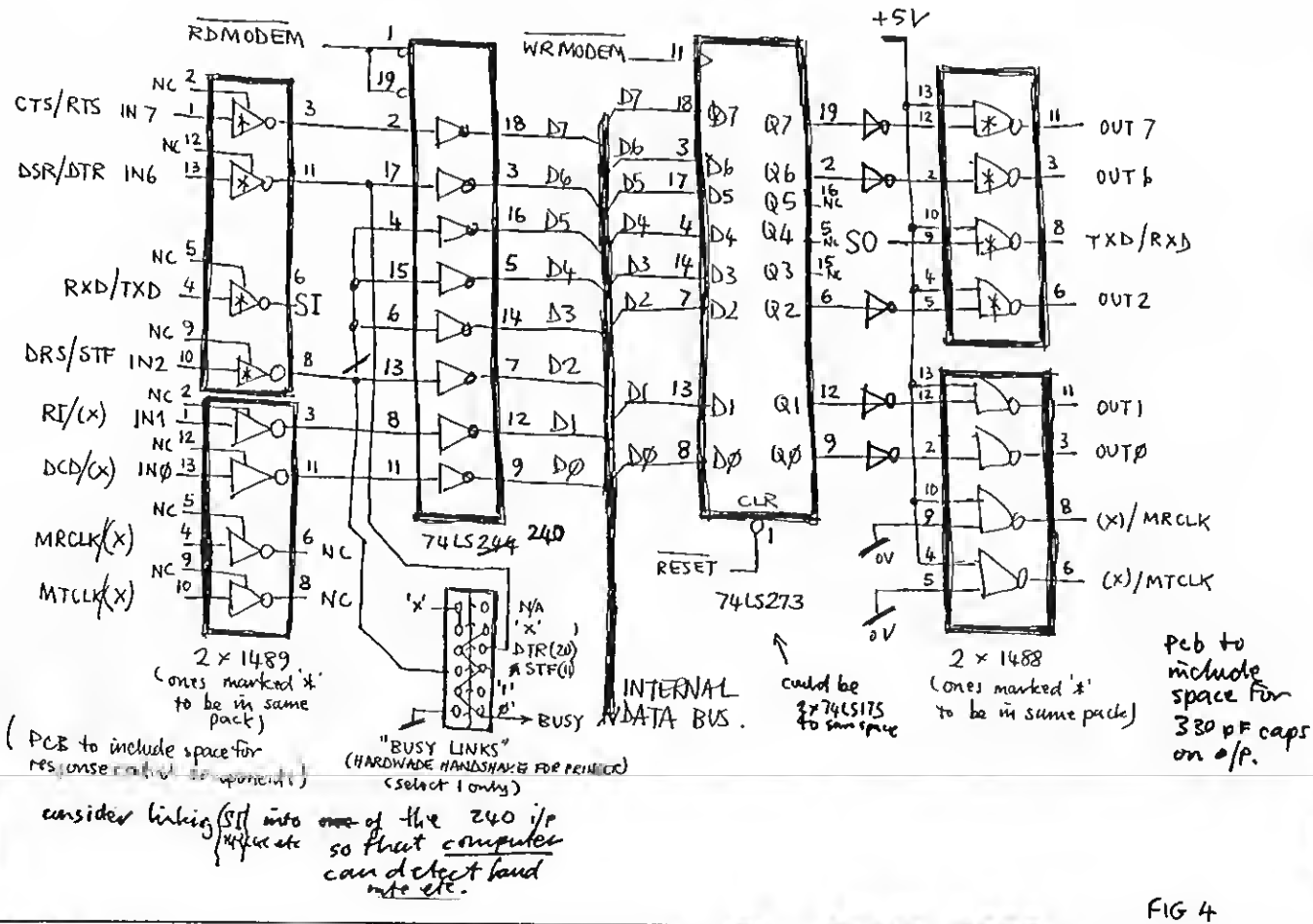


FIG 4

COM 1 CIRCUIT DIAGRAM (Figs 5 and 6)

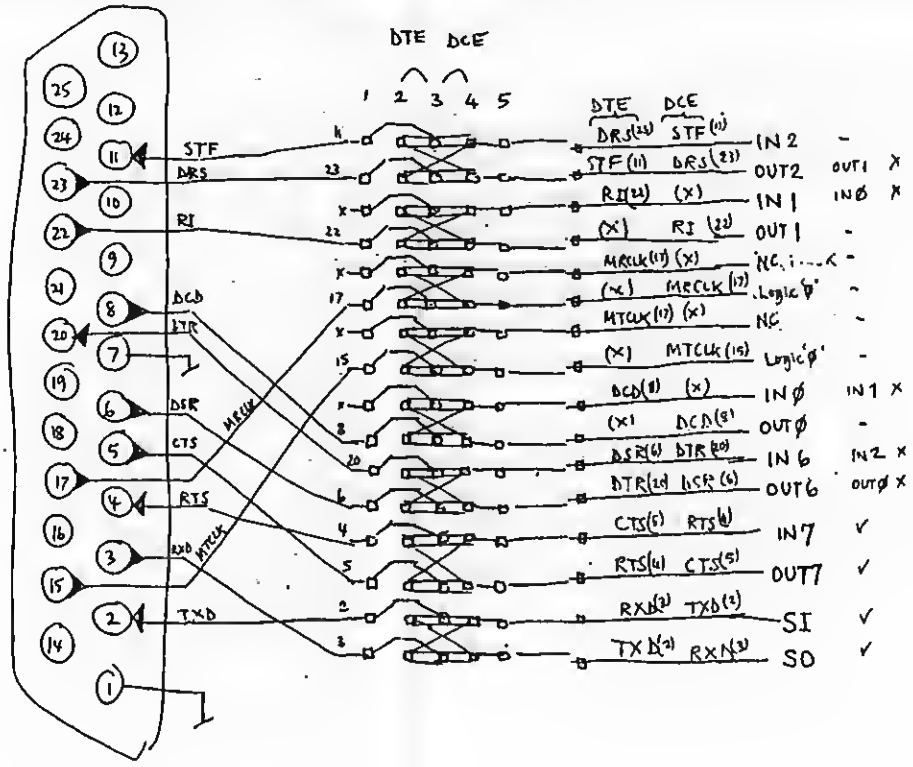
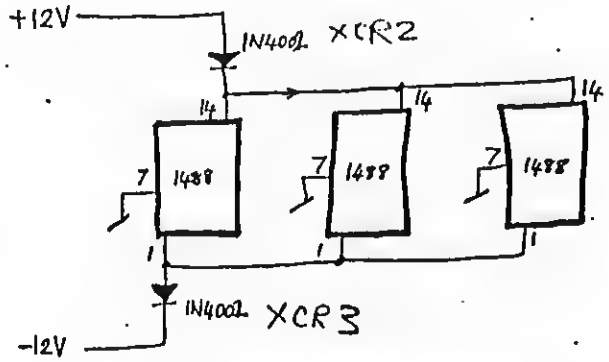
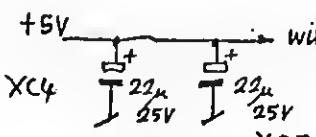


FIG 5



Protection diodes for 1488 must be fitted.

1489's - leave response controls open. (but make suitable arrgts on pcb to trim response)



with 100nF decouplers (at least one per 2 chips)
10 off DEC
XC6, 7, 8, 9, 10
11, 12, 13, 14, 15

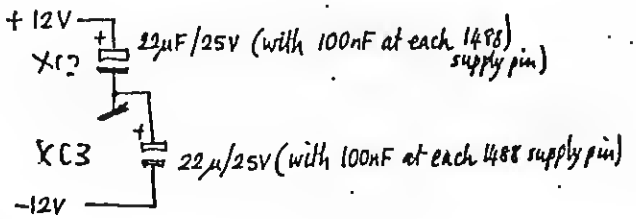


FIG 6

COM 1 CIRCUIT DESCRIPTION

The circuit description will be given with reference to the fig's of the circuit diagram..

Fig 1 (Port Address Decoding)
Fig 1 shows the port address decoding. The card responds to a block of 4 consecutive I/B addresses, which are set by making the appropriate settings on the upper 6 switches of DiL switch XS1 (the remaining 2 switches of XS1 do not influence the addresses). Leaving the switch off corresponds to a '1' bit in the chosen port address; making the switch on corresponds to a '0'. When any of the selected group of 4 ports is accessed by the computer, pin 19 of the 74LS688 comparator goes low enabling one or other of the two halves of the 74LS139. if it is a Read ("input") port access, the upper half of the '139 is activated; if it is a Write ("Output") it is the lower half.

The allocations for the Read addresses are as follows: (Note that the read and write ports are expressed counting from a base of 0, but if the address selection links are set to choose any other base then this must be added, for example if the switches are set to 2B (hexadecimal notation) the following x0,x1,x2,x3 ports will become 2B,21,22,23.)

x0 Read the UART Status (to see if data can be written to or read from its data port).

x1 Read the UART Data Port (do this only after you have checked the UART status).

x2 Read the various control input lines of the modem.

x3 Not used; but if there is space on the circuit board this will allow the computer to read a DiL switch so that the user can enter various default settings, to be adopted by the program which sets up the board.

The 4 write ports have the following significance:

x0 Writing to this port sets the various operating parameters of the UART (for example the number of stop bits, parity etc).

x1 Write the data byte to be transmitted next to this port (do this only after you have checked the UART status by reading Port 0).

x2 Set or reset the various control output lines of the modem.

x3 Writing to this port sets the baud rate for receive and transmit. The upper 4 bits select one of 16 possible baud rates for transmit, and the lower 4 bits select one of 16 possible baud rates for receive. Often the receive and transmit baud rates are chosen to be the same, but in many systems different rates are chosen for receive and transmit, and it is a great weakness shown in inferior designs to insist that both rates are to be the same.

Fig 2 (Clock Gen, Data Bus Tranceiver)
Fig 2 of the circuit diagram shows the baud rate generator chip "AY-5-0116/COM 0116" and gives the table of available baud rates. if +12V is connected (as shown dotted) a similar chip "BFR 1941L" can also be used. (No harm will be done if this +12V is connected to pin 9 of the 0116, although it does not require this voltage for its own operation).

Also on this fig is the 74LS645 transceiver which buffers all the data in and out. It has two control lines: BUFFEN(L) which enables the tranceiver for the selected group of 4 i/O Ports, and DiR(L) which is low whenever the operation is a read (input), and is high for a write (output). A novel method of generating the buffer enable and direction controls for the 74LS645 has been implemented here: The direction is left in whatever state was specified by the start of the current access (thus providing valuable additional "data hold time" often denied by other arrangements). When the present access is a "write" the enable signal is active continously until the next "read" access. (For a read the enable signal persists only for the duration of the microprocessor read since the MPU timing does not require a data hold time, which is fortunate because it would not be a good idea at all to leave a peripheral card data tranceiver like this enabled permanently onto the systee data bus in the read direction!)

Fig 3 (UART, Power On Reset, Busy)
Fig 3 shows the UART itself. This can be any of the well known "industry standard" 40-pin types, of which the AY-3-1015 shown here is typical. Extensive use has been made of dedicated RS-232 interface chips to translate the RS-232 levels to and from the TTL levels used by the rest of the computer. The 1488 (also known as 75188) is used to drive a line to RS-232 levels, and the 1489 (also known as 75189) is used to receive those levels and translate them back to normal TTL. if there is space on the board we may make arrangements to allow for "true" or "inverted" serial signals, although this is hardly necessary because the convention of data transmission at RS-232 levels of "mark" and "space" is well established and should not need inversion (unless someone else has dropped a clanger).

Certain UART chips do not initialise correctly unless they are physically reset by a "reset" signal (this sometimes explains why some people insist that a more expensive version of the UART chip gives more reliable operation than the "ordinary" types - perhaps the only difference is the "power on" behaviour.) A "power on reset" is provided by the 10uF capacitor XC1 (the diode XCR1 is used to discharge this quickly at power down so that it will be ready to provide the reset pulse again should power be restored immediately), and the NRST signal from the bus is also brought into the circuit so that this card will be reset whenever the rest of the coputer is reset (usually by operating the reset switch on the CPU card). (An option to be considered for the design of the pcb is to allow the NRST path for reset to be disabled; during testing of new software it can be quite useful if pre-established settings of UART parameters and modem control lines are not wiped out each time the CPU reset button is pressed. This will not be presented as a user selectable Jumper link setting for the modem control lines, because it is usually desirable that they are returned to known states on reset, to guarantee that the modem releases the telephone line, and prevent accidental massive telephone bills.)

in normal operation the RDSTAT(L) line is brought low to allow the UART status to be gated on to the internal data bus on this card and so on to the main computer data bus. The main status signals to be examined are :-

"TBMT" (transmit buffer empty)
"DAV" (data available),

These have to be true before you can send or receive data respectively from the UART, but other signals of interest are :-

EOC (end of conversion),
OR (overrun),
FE (framing error)
PE (parity error).

Special arrangements have been made (Heitron fashion) in the route of the signal from pin 22 of the UART - TBMT. Even if the transmit buffer is

indeed empty, the computer will only believe this if the "BUSY" input is in the state which indicates not busy. Various link settings (to be described) allow you to choose which state this will be, or if you wish to disable this feature altogether. The idea of the busy line is to provide a very crude method of allowing the state of a printer (when it is "busy", ie unable to receive more data for the time being) to overrule the "empty" status of the UART, and so prevent further transmission of data until the printer is no longer busy.

There are much more sophisticated methods of achieving this (eg X/ON X/OFF or ETX/ACK signalling, or the use of the control lines RTS, CTS etc), but this crude method is very effective as a quick way of achieving the desired results during the testing stages of new or temporary installations, and in the case of a printer connection it is always comforting to know that no matter what goes wrong with the software you may have written, the hardware busy signal will prevent printer buffer overrun. The polarity of the BUSY signal is selectable to suit various special circumstances (more on this in the description of the "Busy Links" on Fig 4. (A future redraw of the diagrams should put the Busy inversion link on Fig 3 so that everything to do with Busy can be presented at one sitting.) The arrangement we intend as normal is to include the inverter, by linking pins 1 and 2.

When data are to be read from the UART (after testing the "DAV" status) the RDDAT(L) line goes low, (when the computer reads that port), and gates the DB0-DB7 data lines on to the bus. (Tri-state gating is used, so that no conflicts arise.) Notice that the "RDE" (read data enable) pin 4 of the UART is also connected to the "RDAV" (reset the data available flag) pin 18 so that the computer can easily be programmed (by testing the DAV flag before each read) not to read the UART data port again until fresh data is available.

The WRDAT(L) line writes data from the computer into the UART (the program having first checked the "TBMT" status) for transmission.

Logically there is no meaning to defining a WRSTAT(L) (write to UART status) port, because the status lines from the UART are developed internally and cannot be written to. However in this design the WRSTAT(L) line serves another purpose: it is connected via an inverter to the "CS" (control strobe) input pin 34 of the UART, and passes the data on the data bus at that time into the control register inside the UART to specify such things as the number of stop bits to be used, parity, etc.

Fig 4 (Modem Control Input Buff & Output Latch)

Fig 4 of the circuit diagram shows the use of the WRMODEM(L) and RDMODEM(L) ports. After a write (output) which takes WRMODEM(L) low the 74LS273 latch will store whatever data was written to it. This can be whatever is required to control the modem in use. Similarly any of the signals from the modem can be read (input) at the RDMODEM(L) port.

A buffer for 8 input lines and a latch for 8 output lines have been provided, but not all 8 of the output lines have been used.

The SI (UART Serial data input, from RXD or TXD on the D connector) has been brought into the input buffer, bit 5, so that the software has the opportunity of examining this line directly to attempt to determine the incoming baud rate, and set the UART accordingly. Some of the "intelligent" modems have built in means for doing this, but if the device was something other than a modem - an unintelligent serial keyboard, say - the intelligence quotient of a modem would be irrelevant. (The crude way of determining an unknown baud rate is simply to try all 16 rates and examine the UART output, plus its framing error, parity error, etc, outputs until sense is detected in the gibberish. Of course the connection we are discussing for SI does not prevent the "suck it and see" approach from being adopted if you prefer.)

Two further signals, MTCLK and MRCLK, are shown connected to the input buffer. These are the extra clocks (Transmit and Receive, to data bits 4 and 3 respectively) which are output by high speed modems. They are of little direct use connected to the buffer (save to allow the software to test these bits to determine that a high speed modem is present), and their main purpose is to allow a convenient point for connecting some unknown signal from the 25-way D connector to the buffer if required. (This is done by replacing the appropriate jumper link(s), shown on Fig 5 of the diagram, with a direct connection to the signal of your choice. I cannot be more specific here, because I can't think of a reason why you would want to be doing this, I have merely provided the facility to let you make alterations if you wish.)

There is some more to be said on the topic of the "busy" line. At the lower left of the diagram on Fig 4 is a set of links called "Busy Links"; these allow various options as follows (note that on the finished pcb we intend to alter these links so that "X" is at the bottom and the 8V connection is at the top). As the present drawing stands "1-1" is at the top and "6-6" is at the bottom.

Link Function

1-1 Not used (the left-hand pin 1 is merely a termination point for some special signal the user wants as busy: signal "X")

2-2 Signal "X" is the busy signal.

3-3 DTR (RS-232 pin 20) is the busy signal (for a COMB-1 board configured as DTE, connected to a printer also configured as DTE, the conventional "crockpot" way). This suits many printers, eg Epson with B148 and some other serial RS-232 interface, Qume Sprint 5 Daisywheel etc.

4-4 RTF (RS-232 pin 11) is the busy signal (for a COMB-1 board configured as DTE, connected to a printer also configured as DTE, the conventional "crockpot" way). This also suits many printers, eg Epson with B148 interface.

5-5 This is the selection of a permanent "busy" signal. On the face of it of little use, but perhaps of benefit for a switch to prevent the transmission of data for some reason (eg to prevent damage to a repair man who is reclining inside a printer mechanism which is powered on for some test purpose.)

6-6 This is the selection to disable the "busy" signal, ie no busy signal can take effect. This is useful as a test position for a printer which has a software handshaking protocol implemented (eg XON/XOFF) - so that you can test that it is the software performing the handshaking, not the hardware busy signal. And of course this is the position for those applications where a printer "busy" has no meaning, for example if the COMS-1 is the interface to a Modem, its primary application.

Normally: link 6-6 and vary this to introduce a busy signal only if circumstances dictate that one is needed.

(The strange zigzag layout of the common signal to the group of pins has been adopted to minimise the chance of detrimental short circuits if a jumper link is inadvertently placed in a vertical position, through force of habit, or momentary inattention on the part of the user.)

The RS-232 to TTL, and TTL to RS-232 level translating ics (i489 and i488 respectively) have an inverting action, ie a rising voltage on their inputs produces a falling voltage on their outputs and vice versa. This is in agreement with the convention of "negative" logic already discussed for RS-232, and explains why no compensating additional inverter is normally required on the SI (serial in) and SO (serial out) lines, ie RXD and TXD. However we ordinary people like to think in ordinary "positive" logic levels, where +5V is a logic "1" or "high", and 0V is a logic "0" or "low". Therefore the input buffer we use is a 74LS240 (an inverting device) and similarly the outputs of the 74LS273 are additionally inverted by a further 74LS240. The additional inversion in the 74LS240 effectively cancels out the inversion in the RS-232 level translation ics. The particular choice of the 74LS240 for this purpose has been made so that any users who dispute the wisdom of using positive logic in this application are at liberty to substitute a 74LS244 in either position. (We must note that spare packages in these chips must therefore not be used as inverters in the finished pcb design.)

There however is one very significant resulting benefit of the inverters after the 74LS273: At power on, or when the CPU reset is operated (if this function is enabled of course) the 74LS273 initially has outputs which are all "0"s. this means that the D connector signals listed next are all at the negative voltage level, which is defined as the "OFF" condition:

D Connector	Sig Name	State Pin	(DTE connection)
4	RTS	OFF	(ie no request)
11	STF	OFF	(ie "Originate)
20	DTR	OFF	(ie not ready)

The most important signal above which I want to mention is DTR (data terminal ready). It is often vital with some types of "auto answer" modems that this signal be OFF at power on, until the operating software explicitly turns it on. (Otherwise, certain manufacturers modems will take it on themselves to answer any incoming call at the first incoming ring, automatically select the baud rate, and then be totally impotent to do more, as no operating software is yet in control.) And obviously it is in general a contradiction in terms to issue the important DTR (Data Terminal Ready) signal to a modem if in fact the Data Terminal is not ready.

A further benefit to choosing the DTR polarity to be OFF at power on and reset, is that if some communications experiments you are making get into a mess, (for example if you are trying to make the modem disconnect from the telephone line, but thanks to an error you have made in your software design the modem is not able to receive your commands), the Interak computer system can automatically force DTR OFF if the computer reset switch is activated. (DTR OFF is taken by most modems as a clear hint that they should disconnect from the telephone line forthwith .)

(I must here express my thanks to users Andrew Dawson-Haddocke, Tom Evans, and Dave ('Flash') Gordon who offered much helpful discussion on these matters whilst the COMS-1 design was being devised.)

The 1489 and i488 devices were designed by their manufacturers to have specific features that are beneficial when they are applied to their main purpose of connecting to the RS-232 interface. Firstly (with suitable protection diodes where necessary, built into this design of course) they are not damaged if computer power is removed when they are connected to an active RS-232 line. The 1488 for example offers at least 300 ohms to such a line under all conditions (which is much preferable to a short circuit, which might be the result of some home-made discrete circuit or casual op-amp bodge).

Nor are they damaged when they are active, but the RS-232 line is inactive. Thus it is not necessary to worry about the RS-232 connection when switching on and off mains supplies to the various peripheral equipment (printers, modems, etc) to which the COMS-1 board may be interfaced.

A further benefit of the 1489 and 1488 devices is that they default to known levels when the power is down at either end of the link (there is provision for altering the input thresholds of the 1489 by means of external components, but in this design we find the thresholds as given are perfectly acceptable for our purpose.)

This is how the 1489 device interprets some of the possible inputs presented to it:

Input	Interpreted as:
+12V	ON
-12V	OFF
0V (and open circuit, or "dead" line)	OFF

The 1488 device will present the following outputs for the following conditions:

Output State	Voltage	(Will be interpreted at receiver as):
ON	+12V	ON
OFF	-12V	OFF
Unpowered	0V	OFF

You will see that the state of power down at either end (sending or receiving) of the RS-232 line is to default to the "OFF" condition, thus for example a modem will see its DTR (data terminal ready) input "OFF" if the computer is powered down or reset.

(It is at this stage that we see that our earlier distain at the confusing negative logic levels used for RS-232 was not fully justified; there was definite method in the old timers' madness - they'd thought of everything before the home and small business computer lobby came in and messed things up')

Each driver in the 1489 devices has a "response" pin. This allows the input threshold voltages to be adjusted by a resistor to +12V (say) or -12V, and the ac (alternating current, or switching) response to be adjusted with a capacitor to 0V. The polarity of the voltage chosen to which to connect the resistor will be influenced by what logic level you want an open circuit input to assume. (As I said before, the existing thresholds of the devices "as supplied" are perfectly acceptable to us, thanks to the addition of the 74LS240 inverters immediately after them, but we intend the finished pcb to have provision for small capacitors to allow the ac response to be varied.

Typical arrangements have a 47k resistor from the response pin to -12V, and a inF capacitor from the response pin to 0V, but in the absence of better information you can begin by leaving the response pin open circuit. (I recommend you leave things alone unless you know what you are doing, in which case you will need no guidance from me!)
In a similar manner the output ac characteristics of the i488a can be tailored for particular circumstances. The speed of the transition from one logic level to another can be reduced by adding a capacitor from each individual 1488 output to 0V. The benefits of this are to reduce interference and crosstalk along the RS-232C connection. Such undesirable effects are aggravated by fast rise and fall times, and in any event the RS-232 standard probably has something to say on this subject. A typical value for the capacitors in these positions if you want to try them is 330 pF, but I would not blame you if you went along with the majority and simply left them out!

Fig 5 (D Type Connector and DTE/DCE Selection)
Fig 5 of the diagram shows the connections to the 25-way D type connector and the DTE/DCE jumpering arrangements. You will remember that the CDMS-1 board can be used either as DTE or DCE (and also in a hybrid manner as a DTE with some of the connections crossed over, to save users the inconvenience of producing special crossed leads and using breakout boxes for knotty problems). Therefore the lines leaving this diagram each carry two names, eg DRS/STF, according to the positions of the DTE/DCE links on the group of 16 5-pole pin assemblies ("X" means there is no signal allocation for that direction on that particular line). If the possibility of external crossed-over cables may be ignored for the moment, the signals on the RS-232 D type connector itself bear defined, unchanging names, and their direction is defined according to whether this board is configured as DTE or DCE.

Rigorously, this board should have a male connector if it is to be thought of as a DTE, and a female connector if it is to be thought of as a DCE. However we recognise that it is too late for anyone to impose rigor on a situation which has careered entirely out of hand, and we would not think too badly of you if you chose to use a female connector in every circumstance. (However note that nobody we have heard of so far ever uses a male connector on a DCE; DCEs always have female connectors so you are definitely on your own if you configure a DCE with an (incorrect) male connector).

The "DTE" allocations are those which obtain when pins 2-3 are linked on each pin assembly, and the "DCE" allocations correspond to links 3-4. The pins 1 and pins 5 are merely test or termination points, and allow caution to be thrown utterly to the winds if you wish to undertake free form knitting to scramble the wires entirely (perhaps to suit some unyielding third party equipment's requirements).

Note that a common piece of jumpering which is needed for some DTE applications to force action from a set up which doesn't want to work with normal connections (don't ask me to explain why, I think the whole system has been overrun by madmen) is to connect DSR (D connector pin 6) to DTR (D connector pin 20) and DCD (D connector pin 8). If you need to make such a connection, you will be pleased to see that on the group of 16 pin assemblies these signals are very close to one another and may be connected by wire wrapping to pins 1 or 5 or others as appropriate, or even by vertically positioned push fit JLinks for some simple configurations.

In similar vein note also that RTS (D connector pin 4) and CTS (D connector pin 5) are adjacent, and may be shorted together if necessary to suit some particular system requirements.

Fig 6 (Power Supplies, Protection)
Fig 6 of the circuit diagram gives some power supply details, and mentions the need for decoupling, both with electrolytic capacitors to provide a local reservoir of charge, and high frequency decoupling capacitors to cope with transient requirements when logic levels switch. Protection diodes have been inserted into the supply lines of the 1488 devices so that when the computer is powered down there will be no conduction path from any of the lines connected to the outputs of the 1488s. If this is done no damage will occur to the CDMS-1 board outputs or inputs even if it is connected to an RS-232 line which is being driven with voltages from the remote end. This subject, and that of tailoring the response of the 1488, has already been covered in the description of the circuit on Fig 4 of the diagram.

Specific guidance of settings for use with recommended WS 4000 modem, and Interak Communications Software now follow :-

It is suggested that you locate this card starting at 20H (the "H" means hexadecimal notation). The DIL switch settings for this are (see Circuit Diagram Fig 1):

S1-1 ON Bit 7=0
S1-2 ON Bit 6=0
S1-3 OFF Bit 5=1
S1-4 ON Bit 4=0
S1-5 ON Bit 3=0
S1-6 ON Bit 2=0
S1-7 ON Bit 1=0 (these last two are "Don't Care"
S1-8 ON Bit 0=0 settings, set ON for tidyness)

The "Invert Busy" jumper (Circuit Diagram Fig 4) should be set to link pins 1 and 2.

The "Busy Link" (Diagram Fig 4) should be across 6-6, thus disabling the Busy function entirely (nowadays all handshaking with modems is generally executed by the software).

DTE/DCE selection links (Diagram Fig 5) should all (16 in total) be placed in the "2-3" positions to configure the CDMS-1 board for DTE operation.

Communications Software for CDMS-1
There is no need to reinvent the wheel and write your own software for this purpose. A useful public domain program, which is reasonably easy to alter to suit your own particular set up is "UKM7" from the CP/M user group and others.

Better still, it is available already partially patched from the SYSOPs at the Interak Bulletin boards (or from Greenbank Electronics in case of difficulty). (On a 3.5" Interak CP/M diskette the charge from Greenbank is 5.00 + VAT, including the diskette itself, but a friendly SYSOP may have it cheaper, or even let you download it for free if you can accept it that way - don't forget that you can still call a Bulletin Board and download the initial software even if you only have a tape system operational at the time.)

Also see IUGN 14 for the COMM driver software that will allow you to run a modem using a Zymon tape based Interak.

Bulletin Board Contacts:

(Tom Evans)
TAECOMM-INTERAK BB
129 Cranborne Waye
Hayes
Middlesex
UB4 0HR
Data: 01-573 0822
Times (1900-0700 hrs)
Voice: 01-561 2639

(Dave Gordon)
FLASH GORDON'S BB
The Planet Mars
c/o 229 Stonelow Road
Dronfield, Derbyshire
S18 6ER
Data 0246-410073
Times (2300-0700 hrs)

However to aid new users patching existing programs, or even writing their own here are some brief notes of software guidance:
The various procedures are very easy, and only take a few lines of program. Note that this is in stark contrast to the convoluted code required to work many of the dedicated microprocessor "family" SIO, DART, USART, etc chips; they can look simple on a circuit diagram, but are much harder to use in practice (and anyway do not always include all the modem etc control lines presented in this design).

Before you begin you should perform the following initialisation:

1. Output to Port WRSTAT(L) a suitable pattern of bits for the control register inside the UART. In this design as given the bits have the following significance:

D7 The etate of this line is ieeaterial.

D6 The state of thie line is ieeaterial.

D5 NP (no parity). If aet (= "1") will add no parity bit will be appended to the data bits. If reset ("0") will append a parity bit.

D4 EPS (even parity select). If a parity bit is to be appended to the data (see under NP above) it will have even parity if EPB is "1", and odd if EPS is "0"

D3, D2 NB2, NB1 respectively. These control the number of bits to be transmitted, according to the following table:

NB2	NB1	
0	0	5 bits
0	1	6 5its
1	0	7 bits
1	1	8 bite

D1 The state of this line is ieeaterial.

D0 TS0 (transeitted stop bits). This selects the number of stop bits:
0: 1 atop blt
i: 2 stop bita for 6,7, or 8 bit data, snd 1.5 stop bits for 5 blt data. (as selected by NP2 and NP1 above).

For 8-bit, no parity operation (as comeonly used on bulletin boards) the pattern of bits is 0FFH. In Z80 assembly language:

```
;UART set up
UARTSU: LD A,(0FFN)
        OUT (STAT),A
```

2. Output to Port WRBAUD(L) an appropriate pstttern of bits to set the chosen baud rates for transmit and receive. The upper 4 bits of the byte select the baud rate for transmit, and the lower 4 bits select that for receive, according to the following tsble :-

-------Transeit------						------Receive--------					
D7	D6	D5	D4	Hex.	Baud	D3	D2	D1	D0	Hex.	Baud
0	0	0	0	0	50	0	0	0	0	0	50
0	0	0	1	1	75	0	0	0	1	1	75
0	0	1	0	2	110	0	0	1	0	2	110
0	0	1	1	3	134.5	0	0	1	1	3	134.5
0	1	0	0	4	150	0	1	0	0	4	150
0	1	0	1	5	300	0	1	0	1	5	300
0	1	1	0	6	600	0	1	1	0	6	600
0	1	1	1	7	1200	0	1	1	1	7	1200
1	0	0	0	8	1800	1	0	0	0	8	1800
1	0	0	1	9	2000	1	0	0	1	9	2000
1	0	1	0	A	2400	1	0	1	0	A	2400
1	0	1	1	B	3600	1	0	1	1	B	3600
1	1	0	0	C	4800	1	1	0	0	C	4800
1	1	0	1	D	7200	1	1	0	1	D	7200
1	1	1	0	E	9600	1	1	1	0	E	9600
1	1	1	1	F	19200	1	1	1	1	F	19200

For 300 baud receive and transmit (often used on bulletin boards) the pattern of bits is 55H. In Z00 assembly lsnguage:

```
;Baud rate set up
BAUDSU: LD A,(55H)
        OUT (BAUD),A
```

3. Begin MODEM operations. DTR (Data Tarminal Ready) and RTS (Request To Send) which were previously DFF by the power on reset mechanism are now both to be turned on. In Z80 assembly language:

```
;Activste MODEM
MODEMA: LD A,(20N)
        OUT (MODEM),A
```

4. You now have to send a sequence of commands to the modem to sat it up internally to suit specific requirements, eg disable/enable auto anewer, redefine escape saquences, etc, etc, (see eodem manufacturers manual for guidsnce). Then you have to issue soma dial commands (assuming an sutodial modem such ss the WS4000).

All transeission snd recaption, of commands or dsta, is sent vis the UART. The general wsy of outputting data to the UART is ss follows:

```
;Output (send s byta)
SEND:   IN A,(STAT)  ;UART Status Port
        AND 00H      ;Test TBMT (bit 7)
        JP Z,SEND    ;Wait for TBMT=1
        LD A,D       ;Assuming Dsta is in D
        OUT (DATA),A ;Send data
```

The receipt of data or commands is similar:

```
;Input (receive a byte)
RECEIVE: IN A,(STAT)  ;UART  Status Port
         AND 40H      ;Test DAV (bit 6)
         JP Z,RECEIVE ;Wsit for DAV=1
         IN (DATA),A  ;Receive  data in A
```

These routines will ioop for ever in the event of some unexpacted hardware conditions, so a more sophisticated eet of routines would have to decide whst to do in this event, and/or allow some sort of escape (eg a key pressed by tha user) to break out of an endless loop. Also there are some UART status linea which can be checked if desired, eg PE (Parity Error), OR (Overrun Error), FE (Framing Error). This is only from the point of view of the UART of course - it only has the most basic chacks available to it so even if the UART reckons it has transmitted the data OK this is not a guarantee thst the remote party received it.

The program can of course be as complicated as you choose to make it, but I have tried to keep it simple for the benefit of the user who really needs tha help, ie the user doing this for the first time.

5. A very useful signal which can be testad before sending data to a modem is the DCD (Data Carrier Detect) signal. If you wsnt to test this, it csn be done as follows (I have shown this as a subroutine, becsuse DCD should be tested regularly eg in the middle of sending iarge blocks of data. It is upsetting to reach the end of a one hour transmission to discover that the called party hung up half sn hour ago! The loss of the data carrier is a clear indicstion thst the communicstions link is broken snd appropriate action should be taken.)

```
TESTDCD: IN A,(MODEM) ;Modem Port
         AND 01H      ;Test DCD  (bit 0)
         RET
```

On the return from this routine the Zero flag in the Z80 can be tested to see if the Data Carrier was present or not. A non-zaro flag indicstes that DCD wss "ON" (ie carrier present); a zero flag indicates DCD was "DFF" ie (carrier not present).

6. After you have finished, don't forget to hsng up! If you know of no better method you can aiways pull the plug on the telephone iine connection to the modem, or press reset on the computer which will force DTR (Dsts Terminal Ready) "DFF" snd thus the modem to automsticslly "hang up". However common

decency when calling a bulletin board is to follow the specified log-off procedure (often this is just a matter of entering a "D" for Goodbye selection from a menu). Intelligent modems have a command (eg the Hayes command ATH) to instruct them to hang up, but a crude way to force this is as follows, in Z80 assembly language:

```
MDMOFF: XOR A          ;ie set A reg to 00
        OUT (MODEM),A  ;
```

It is generally a good idea constantly monitor the DCD (Data Carrier Detect) line and hang up immediately this is lost, since like it or not, the loss of DCD indicates that communications are at an end. Modern intelligent modess (eg Miracle WS 4000) are also alert in this respect, indeed will even take appropriate action if they detect no transmission of data during a certain "timeout" period; to save an expensive line being held with computers dead at both ends.

Further sophistications in the program are possible with this design, but are left to you to implement. For example you could arrange for the software to sense the baud rate in use, so that it need not fixed at some, perhaps incorrect, preset value. You could use spare inputs and outputs (if any) from the ROMODEM(L) and WRMODEM(L) ports to control peripheral equipsent, eg power supplies, printers, alarm bells. (If so doing note that the outputs and inputs to these ports are now at RS-232 levels.)

D M Parkins 19/9/85, revised 14/10/86

Cossercial Break:

BUY YOUR MODEM TODAY FROM BREENBANK ELECTRONICS!

Intelligent auto answer, auto dial, Hayes Compatible V21/V23. 300 baud full duplex, 1200/75 baud full duplex, 75/1200 full duplex, 1200/1200 baud half duplex. Speed Buffering. Upgradable (in factory) to 2400 baud, and numerous options.

WS4000 £149.50 + VAT

(Further details on request, if not cunningly included with this document)

********** THE END **********

LETTERS

Bob Cowdery (D3UKB),
1 Woodland Way,
Oaklands,
Welwin,
Herts,
AL6 0RZ.

Dear Ed,

Just a short note to bring things up to date. I have now installed the 64 column VDU (official Interak version). This certainly makes the machine more useful and seems to work OK except that I sometimes get some pixels left in the last pixel column of the screen and don't as yet know why.

As i have not received the summer news letter yet I don't know if you have included the Part Screen Handler I submitted back in July sometime. [Ed - this issue]. However, if you have there are a few problems to point out as follows :-

1). For a 64 col screen don't follow my instructions as I did only to find they don't work. Instead change the equate SHIFT to 6. You will also find a bit of bad programming practice in the routine SCRDLL, there are two places where a literal 32 has been used instead of the equate WIDTH. Finally you must change the equate WIDTH to 64 as indicated.

2). A couple of bugs crept in :-
A). The equate MAXCOHT should be 15H not 15, this will cause the bottom of the control table below CR to be ignored.
B). Using the inverted character set, any character with the top bit set will cause a crash. The cure is to insert the instruction ANI 7FH before the CP 020H on line 555.
C). Finally, the comment in CLOSE is incorrect, it should say INPUTS: D=Stream number.

On a different tack, although I obtained a new ZYMOH from Greenbank for the VDU2K I still had the problem of updating ASM32 to ASM64. As I still seem to have no success getting software out of Pete Vella I decided it may well be quicker to have a go myself. This proved to be more difficult than I thought and took an entire Saturday evening and all day Sunday, and there are still a few problems left. However if anybody else is interested here are the mods :-

```
13EF - C0 : 1400 - F5 : 13FD - C0 : 13FE - F5 :
1409 - 3F : 1415 - 05 : 140E - 40 : 1421 - 06 :
13C4 - F5 : 13C3 - C0 :
```

After doing these mods you will find as I did that the tape will not run at 2400 baud any more as the scroll time for the larger screen is now too long. If you want to run at 2400 then the easiest way is to stop the output to the screen when loading. This can be done by the following mod :-

```
12A5 - 00 : 12A6 - 00 : 12A7 - 00
```

Finally the things that this doesn't fix as yet is that the delete key doesn't work properly (mind you it never did on the ASM32 I have either), and on assembly to screen the PAUSE comes up in the middle of the screen and it only scrolls half a screen between pauses.

I've also included another short listing which uses the Part Screen Handler. [ED - Elsewhere this issue]. It's a very simple electronic typewriter. I wrote this mainly because I don't have any word processor software at present and this at least lets me get words onto paper (this letter was written using it). However if you don't have the Part Screen Handler this isn't going to be very much use to you.

Well I think I've rambled on for long enough, so until the next time.

Bob Cowdery (G3UKB).

[Ed - Thank you Bob for a really good article, the study of which will give considerable insights into assembly programming. I think Greenbank do a disk package that includes a word processor with a disk based assembler which I am sure you would find of great interest. Your additional program has been placed at the end of the original as it follows very naturally on from it.]

=======

F.R.Johnson,
32 Langdon road,
Folkstone,
Kent,
CT19 4HX.

Dear Bob,

I have recently aquired an 80 column VDU which has 22 programmable function keys. Unfortunately the programmed data is not retained after switch off and it therefore occurred to me that it would be useful if these keys could be set up automatically on a cold start. I therefore decided to change the normal "DIR" startup command to facilitate this. Using DDT on the file CPM64.COM I found it to be stored at 0A07 in the form :-

```
0A07 03 Length of command
0A08 44 D
0A09 49 I
0A0A 52 R
0A0B 00 End of command
```

There are 15 bytes available which I have used to store the following as a new startup command :-

SUBMIT STARTUP

Using DDT this can be set up as below :-

```
A>DDT CPMxx.COM  Substitute size of your system
                 for xx.
DDT VERS 2.2
NEXT PC
2900 0100
-S0A07
0A07 03 0E
0A08 44 53
0A09 49 55
0A0A 52 42
0A0B 00 4D
0A0C 20 49
0A0D 20 54
0A0E 20 20
0A0F 20 53
0A10 20 54
0A11 20 41
0A12 20 52
0A13 20 54
0A14 20 55
0A15 20 50
0A16 20 00
0A17 20 .

A>SAVE 40 CPMxx.COM
```

Each disk may now be SYSGEN'd with the new version and a suitable STARTUP.SUB file written to set up keys and/or run programe to suit the disk.

F.R.Johnson

[Ed - A very neat and original idea this and it reminds me of the IBM VM Profile mechaniam. One suggestion. Rename SUBMIT.COM as SUB.COM. This as SUB TASK is simpler than SUBMIT TASK for oft repeated operations. Your idea opens up the Interak to all sorts of add on equipment, colour screens etc, as each can be initialised at cold start. I wonder if you could tell us about the 80 col VDU that you are using, is it available to normal mortals ect?, lots of people need a low cost 80 column interface.]

====

Paul M.Nicklin,
109 Devonshire Drive,
Derby,
DE3 5HE

Dear Ed,

I now have an Interak 40k + Zybasic. The assembler ASM32 is very good and I have used it a few times, but there are some functions that I have seen used but can't get to work on my ASM and so I await the next part in the series of articles on using it.

Many people, I fear may be unwilling to type in this program as, with a hex listing that long, errors are bound to occur (I made 20!). Perhaps some bright spark could come up with a printer routine that prints checksums on the line and an input routine that checks each line of code.

While building the computer a few difficulties arose, all but one have been discovered, the unresolved mystery is hidden on the MZB-3 board which refused to do anything until two MZB's were run together, sure and another, working, board. The card has been no trouble since.

The Zybasic 2 tape didn't load first time, but on inspection of the tape recorder a very loose head was revealed, the head could move freely. A large blob of glue repaired the broken part and with slight adjustment of the head the tape loaded perfectly. We have two tape recorders, the better one is a Philips N2235 which now works very well, the other is a Pye SXB923 which will load the Zybasic tape alright, but won't save anything over 1200 baud.

Zybasic is on the whole very good but it lacks features found on lesser machines such as the ZXB1 (the computer that I used before) such as multi dimensioned arrays, for which formulas can be found [Ed - elsewere in this issue]. The lack of trig functions is apparent but the 'Bits of Basic' help here. At first the string handling was unfamiliar, having used the ZXBi's easier system using TO, but again the problem was soon solved. My main difficulty is the absence of PRINT AT, which can't really be simulated.

Like many other Interak owners it is my second computer, my first was a ZX81 which I have had for over four years. The chances are that if Sir Clive had not invented his ZX81 then fewer Interaks would have been sold.

I have recently left school and have a lot of time on my hands and so I have converted som programs for use on Interak BASIC, the two given seem to be linked, the first entitled BIORHYTHM will print a graph of intellect, emotion & vitality. These repeat themselves every 33, 28 & 23 days respectively. The second will print out a calendar for any month since 1572. The program will print calendars for months before that year but they will be wrong. The reason for this is because today's calendar, the Gregorian calendar, is different from the previous Julian calendar.

Also included are formulae for simulating 3D and 4D arrays, by looking at the pattern of formulae for 2, 3 and 4D arrays, higher dimensioned arrays could easily be thought up.

Paul M. Nicklin.

[Ed- Thanks for the programs. I have put them elsewere in this issue. It's wierd about the MZB-3 problem but I'm glad all is now ok. Interak is more flexible that the ZX81 but I do agree it sometimes lags behind, due I think to lack of commercial push. Overall the people I talk to learn more with an Interak than any other computer and that is it's success. As to the bright spark - Why can't it be you?. Try to produce a checksum dumper and it can be presented to the other members via the newsletter. A study of Zymon may help as I produced checksums with it's tape handler.]

=====

Bob Cowdery (D3UKB),
1 Woodland Way,
Daklands,
Welwin,
Herts,
AL6 ORZ.

Dear Ed,

Just a short note to tell you that having completed a basic Interak system, I am now getting quite a bit of enjoyeent froe using it. However I eust sympathiee a little with Mel Saunders point of view as I too have an outstanding order for S/W froe Pete for well over 6 months. I think the thing that is disappointing about it is that there is all this lovely software which is ostensibly available for Interak but aleost impossible to get hold of.

Just to tell you a thing or two about what I ae doing (hello, wheree everyone gone!). I have enclosed e screen driver for VDUK/2K (EO-elsewhere in this issue) which takes a very sieplistic attituds to carving the screen up into a nueber of part screens (I don't think Apple have anything to worry about!). Cynics might say the VDUK or even the VDU2K screen is seall enough ae it is without carving it up into smaller bits, but be that as it may. The source is a bit long for the newsletter running to over 1888 lines including coeeents, thus I think if the editorial collective (ie Bob Eldridge) consider it to be of general interest, then it eay only be a hex duep that is printed. The intsnsion is to develop a sieple minded scheduler for Interak when I get around to building a clock/tieer module, and to use the above screen driver as a part of a general purpose multi-tasking O/S plus whatever else ssems necessary. The ultimate aim of all this (getting to ths point at last) is to provide a suitable operating environment for programs intended for Amateur Radio use (my other hobby). The main problse I am going to run into is the utter tedium of trying to devslop reasonably large prograes without disks. Already, the first bit of it, ie the screen driver gets closs to being too large to fit into memory, end takss soee tiee to load from cassette (I suppose Chrietmas isn't that far away). Anyway enough of my ramblings, I look forward to the next issue of Interaktion which I always find interesting.

Bob Cowdery (D3UKB).

[Ed- Thanks Bob, a very novel idea, diske would extend the range poslbilities by a thousandfold. You could load overlays from disk as required so as to kesp the O/S down to 16k, the rest being for the applications software. Put two disks on and one could hold 708k of O/S overlays loded by the O/S in a form of page mapping. Please keep us inforeed as to the progress as many meebers are interested in this type of advanced application.]

*****

Mike Warton,
"Swevenings",
8 Dvitts Cloee,
Winslow,
Bucks,
MK18 3QD.

Dear Bob,

Many thanks for taking the time to contact ee the other evening regarding the iepleeentation of CP/M on the Interak. In fact there is an error in the listing of the Loader, actually more of an oelssion!

I won't tell you exactly what it is, that would spoil the fun, but look on page 6/7 of IUGN 7. To glvs you a clue, where does the instruction IN) load the disk data to?

Perhaps a note in a future IUGN of the corrected code would prevent any-one else ploughing the same furrow as me.

Regarde and happy bug hunting.

Mike Wharton.

[Ed- Thank you Mike. I am glad I could guide you through it. Actually it wasn't a bug as my Boot Roe loads the HL pair with the correct destination address. Still for a stand alone application the HL pair should be loaded with the data address before the load begins.]

****

Steve Padley,
14 Wickham road,
Farehae,
Hants,
PO16 7EU.

Dear Bob,

In the past I have written a couple of articles and sent software up to the user group. Well this time the ball is on the other foot, I'e after inforeation. Looking back over past IUGN's you seemed to be soeewhat of an expert on disk systems.

Having recently got my disk interface up and running I bought the 3.5" drive from Greenbank (wish I had another already). I now find eyself wondering where to get software for this foreat and size disk. Looking at IUGN 11 I see you mentioned that you were going to send our standard sst up to the CPMUGUK so that they may supply software for us. Does this now eean that ths only possible sources of software will be Greenbank and CPMUGUK !!! My one interest will be obtaining different languages from a disk based asseebler to C, Forth, Cbasic, Pascal and any othsrs. Up to now I've only used Xtal & ASM64. Is it possibls to adapt Xtal to disk or can I get my Basic prograes from tape to diek in soee way?

If you have any addresses of software houses that may provide software for CPM (if they exist apart from CPMUGUK) I would be pleased to have thee. Any assistance you can give would be greatly appreciated.

Steve Padley

[Ed- I am not really a disk expert. I was the first to put my own CP/M onto the Interak, so I had to build a prototype board designed by David, modify the design to eake it work, now known as the FDC1, then construct a DETSYS/PUTSYS/BIOS and BOOT using Zymon with tape cassette. After 3 months sweat and soee blood, CP/M signed on. I learnt a great deal during that tiee; the eost valuable lesson being "Don't be the first to try something." Anyway I have good news for you. The IUGN disk library is operational, see the rear of this issue. Our format was unreadable by ths CPMUGUK and Greenbank have issued a modified format which should let thee reed our disks. Just send them a disk containg some text with a print out of STAT DSK:. I ae puzzled as to why you bother with Forth ect when "C" is the programmers tool at the eoeent? Several versions of "C" including a "C" compiler can be obtained from "Brey Matter Ltd" at 4 Prigg Meadow, Ashburton, Devon, TQ13 7DF. Phone thsm on 0364-53499. Finally thanks for your contributions, they are very interesting to the eeebers, perhaps you could drop me a disk soon with soee more of your work on.]

****

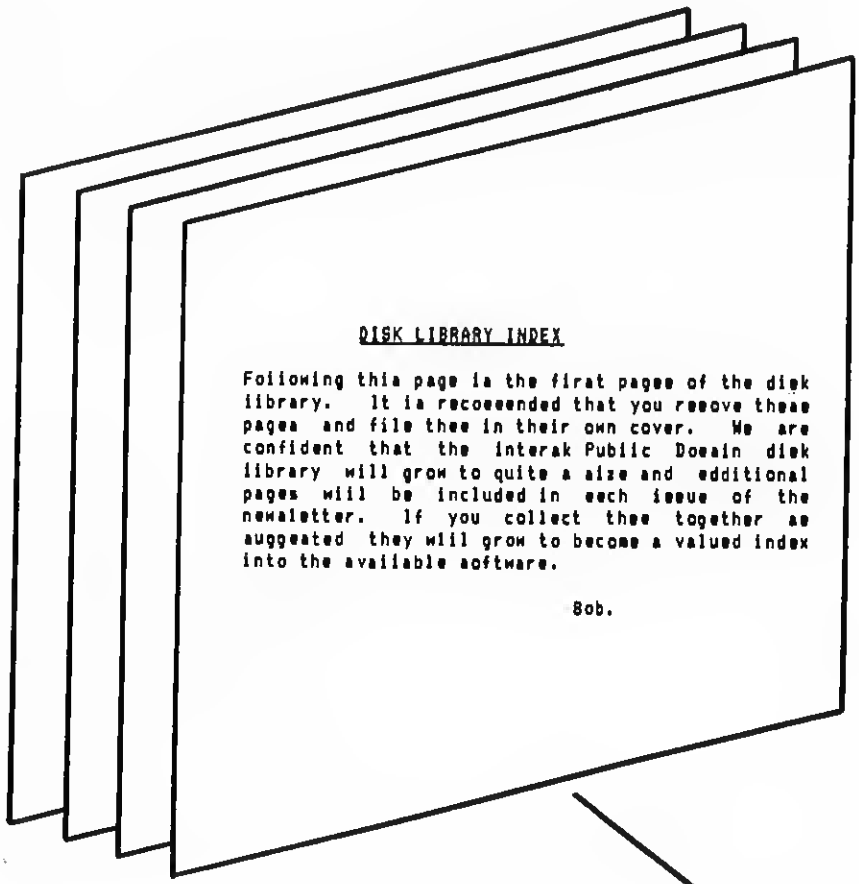
FOR SALE

DTi-i Tape interface card for the Interak, fully populated, but without a front panel. £18.00p or very near offer.
Mike Wharton, "Owevening", 0 Ovitta Close, Winaiow, Buckinghae, MK18 3QD. Tel. Winaiow 4367.

==========

1 x 19" Card cage and case £30.00p
1 x ISOUS with 13 aockets £30.00p
1 x Multi-raii PSU £35.00p
1 x MZO-3 CPU card £20.00p
4 x MXD-2 Dynaeic rae ..... each £2i.00p
1 x DTi-i Tape interface £25.00p
1 x iP-i Opto-input (Keeitron) £20.00p
1 x DP-i Reiay-output (Keeitron) £20.00p
1 x MENTA Z80 aaaeebler training aid ... £70.00p
MENTA is a Z88 development ayatee with TV diapiay, tape i/O, and a keyboard assembler houaed in a A80 caae deaigned aa an aaaeebiy language traininer by Dataman designs of Dorcheater.
Derry Caebeii, 153 Lower Faireead road, Yeovii, Oeeeraet, OA2i 5SR. Tei 0935-70202.

==========



DISK LIBRARY INDEX

Foiiowing thia page ia the firat pagee of the diek iibrary. It ia recoeeended that you reeove theae pagea and file thee in their own cover. We are confident that the interak Public Doeain diek iibrary will grow to quite a aize and additional pages wiii be included in each ieeue of the newaietter. If you collect thee together aa auggeated they wiii grow to become a valued index into the availiable aoftware.

Bob.

INTERAK USER GROUP
PUBLIC DOMAIN
DISK SOFTWARE LIBRARY
INDEX

INTERAK DISK LIBRARY

A public domain software library has been set up for the user group on 3.5 DSDD disks. The majority of the software originates from the UK CP/M User Group public domain library, and I am indebted to David Parkins who provided most of the material.

Contributions to the libary will be gratefully received. These must be original or in the public domain. Programs from magazine listings should not be submitted unless permission to do so has been obtained from the Editor. Items published in the IUGN can be freely given, subject only to the originators permission.
Please give as much information as possible in a seperate .DOC file.

The usual rules apply! Programs may be given away but not resold and copyright notices should not be removed from the programs. As not all the programs have been tested there may be bugs. If you find any please let me know especially if you have also found a solution.

This software index has been designed so that you can collect the pages into a seperate folder. Updated and additional pages will be issued in the newsletter as and when required.

If you have any game for the "Public Domain" send them in and I will make them available to everbody.

How to order a disk volume :-
You must be a member of the Interak User Group.

Charges: Copying charge. 2.00
Media charge. 3.00 or send in your own.

Cheques or Postal Orders (no cash, please) should be crossed and made payable to INTERAKTION and sent to:

Mr C.SRIDGSTOCK,
32 WIMBORNE AVENUE,
THINDWALL,
WIRRAL,
MERSEYSIDE.
L61 7UL.

If you wish to supply your own disks it is advisable to pack them between two pieces of hardboard or similar material as I understand it is possible for them to be cracked in the Post if sent in an envelope or Jiffy Bag without protection.

Library disks are 3.5 inch double sided double density (DSDD) to the Interak soft sectored format.

It is advisable to print the .DOC and .TXT files received with a new volume. This will allow you to fully spreciate the new programs that you have obtained. Also anything named README.xyz is worth printing.

It is probably a good move to copy a disk received from the library. You can then keep the lib copy and gradually collect a full set. If you later require a file from the lib you can find the volume from the index and copy it across to your working disk.

When submitting files to the library please try to stay with the extension names protocol. In general, file extensions have the following meaning :-

$$$	Temporary file.
ASM	Assembler source file.
BAK	Backup file.
BAS	Basic Source code.
C	C language executable file.
CAT	Catalogue file.
CBL	Cobol file.
CMD	DBASE2 command file.
COM	Command file, the executable program.
CRC	Cyclic Redundancy Check file.
DAT	Data file, used by a program
DBF	DBASE2 Data base file.
DOC	Document file, ASCII file describing the other files using the same filename.
HEX	Hexadecimal machine code file. (Loadable).
INT	Intermediate code.
LBR	Library file.
LST	Basic Language source File. ABCII
MAC	MBO source file.
NDX	DBASE2 Index file
OBJ	Object code file. Like .HEX
OVL	Overlay file.
PLI	PL/1 source file.
PRN	Printer listing. Printable ABCII file.
REL	Relocatable module.
SYM	Symbol file.
SUB	Keyboard command file. For SUBMIT.COM
TXT	Text file, ASCII file describing the other files using the same filename.

IUG I - 588k

FILENAME.	TYPE.	SIZE.	REMARKS.
F83	.COM	24K	FORTH-83 FOR CP/M BY PERRY & LAXEN.
README	.88	28K	F83 INSTRUCTIONS. DOC
F83-FIXS	.TXT	8K	F83 VERSION 1.8 UPDATE.
BASIC	.BLK	28K	BASIC COMPILER IN F83.
CLOCK	.BLK	12K	SOURCE FOR A CALENDAR EXAMPLE.
CPU8888	.BLK	44K	8888 DEPENDENT CODE.
EXPAND88	.BLK	8K	ORIGINAL SOURCE TO EXPAND.NUF.
EXTEND88	.BLK	32K	EXTENSIONS SOURCE.
NUFFNAN	.BLK	44K	COMPRESSION PROGRAN.
KERNEL88	.BLK	188K	KERNEL SOURCE.
META88	.BLK	52K	METACOMPILER SOURCE.
UTILITY	.BLK	112K	UTILITY SOURCE.
USQ	.COM	4K	UNSQEEZES SQUEEZED FILES.
LISP	.CON	28K	UPDATED LISP.
INITLISP	----	4K	
INITLISP	.ST8	4K	
LISP	.DOC	16K	INSTRUCTIONS.

F83.CON 24K FORTN-83 BY PERRY & LAXEN.
The following books are helpful in using F83:
Inside F83 by C.H.Ting.
Mastering Forth by Anderson & Tracy.
Forth. A text and reference by Kelly & Spies.

***

META88.BLK 52K METACOMPILER SOURCE.
NVP Meta-compiler that permits the creation of new Forth systems in the Forth language.

***

```
**************************
*        BLANK           *
*                        *
*    FOR LATER USE       *
*                        *
**************************
```

IUG 2 - 548k

FILENAME.	TYPE.	SIZE.	REMARKS.
CINTERP	.COM	16K	COBOL INTERPRETER.
COBOL	.COM	16K	N.P.B. MICRO COBOL VER 2.1
EXEC	.COM	8K	" " " "
PART 2	.COM	16K	" " " "
COBOL	.DOC	48K	INSTRUCTIONS.
ADD	.CBL	4K	COBOL PROGS. SEE COBOL.DOC
CBL1	.CBL	4K	" " " " "
CBL2	.CBL	4K	" " " " "
DEMO	.CBL	4K	" " " " "
SEQ	.CBL	4K	" " " " "
DEMO	.CIN	4K	" " " " "
DEMO	.LST	4K	" " " " "
CBL1	.CIN	4K	" " " " "
CBL1	.LST	4K	" " " " "
CBL2	.CIN	4K	" " " " "
CBL2	.LST	4K	" " " " "
CBL1	.FIL	4K	" " " " "
ADD	.CIN	4K	" " " " "
ADD	.LST	4K	" " " " "
ZSC-1	.C	28K	SMALL-C COMPILER.
ZSC-2	.C	28K	SMALL-C COMPILER.
ZSC-COMP	.LIB	12K	" "
ZSMALL	.COM	24K	" "
ZSMALL	.DOC	16K	INSTRUCTIONS.
C-UTIL	.DOC	4K	DOC. ON TAB, LIST, FILECHOP, AND UNLOAD.
CONIO	.LIB	8K	SMALL-C LIBARIES.
CRUN	.LIB	8K	" "
CZMON	.C	12K	SAMPLE SMALL-C PROGRAM.
CZMON	.COM	8K	" " "
FILE	.LIB	12K	SMALL-C LIBARY.
FILECHOP	.C	4K	CHOP LARGE FILES INTO SECTIONS.
FILECHOP	.COM	4K	
LIST	.C	4K	SAMPLE SMALL-C PROGRAM.
LIST	.COM	4K	
NUMIO	.LIB	4K	SMALL-C LIBARY.
TAB	.C	4K	SAMPLE SMALL-C PROGRAM.
TAB	.COM	4K	
Z80ASMUK	.COM	12K	Z80 ASSEMBLER FOR USE WITH SMALL-C.
Z80DOCUK	.DOC	8K	INSTRUCTIONS.
ADV	.COM	36K	EXPANDED ADVENTURE GAME.
ADVT	.DOC	4K	INSTRUCTIONS.
ADV1	.DAT	32K	
ADV1	.PTR	4K	
ADVT	.DAT	188K	
ADVT	.PTR	16K	

CZMON.C 12K SAMPLE SMALL-C PROGRAM.
CZMON.COM 8K
A Z80 monitor program allowing Set breakpoint, Copy memory, Display disk dir, Execute progras, Input/Output to a port, Read and Write files plus many other facilities.

***

LIST.C 4K SAMPLE SMALL-C PROGRAM.
LIST.COM 4K
List an ASCII file to the console, 20 lines per keypress.

***

TAB.C 4K SAMPLE SMALL-C PROGRAM.
TAB.COM 4K
Converts Intel format files such as .HEX files into a tabulated hex listing. The output can be directed into another file.

***

```
**************************
*        BLANK           *
*                        *
*    FOR LATER USE       *
*                        *
**************************
```

IUG 3. - 568k

FILENAME.	TYPE.	SIZE.	REMARKS.
BCKUP	.COM	4K	DISK BACKUP PROGRAM.
COMPARE	.COM	4K	BINARY FILE COMPARISON.
COPY	.COM	4K	DISK COPY PROGRAM.
CPACK	.DOC	12K	DOCUMENTATION FOR BCKUP, COMPARE, RESTORE, COPY AND SORTDIR.
DDS	.COM	4K	FINDS ADDRESS OF CCP/BDOS & SIZE OF TPA.
RESTORE	.COM	4K	RESTORES ERASED FILES.
SORTDIR	.COM	4K	SORTED DIRECTORY PROGRAM.
DUTIL	.COM	12K	REVISION OF DISK UTILITY WITH EXTENDED FEATURES.
XLATE2	.COM	8K	TRANSLATES INTEL 8080 SOURCE TO ZILOG Z80 CODE.
DIRSCAN	.COM	8K	SCANS DIRECTORY.
INDEXER	.COM	20K	CREATES AN INDEX FOR A BOOK OR ANY DOCUMENT
INDEXER	.SUB	4K	AUTOMATICALLY. INCLUDES A SAMPLE PROGRAM.
PIKEY	.INX	4K	
PIKEY	.TRE	4K	
INDEXER	.DOC	16K	DOCUMENTATION.
DKI	.COM	8K	MENU FILE TO SEND CODE TO DKIDATA 82/83.
EPSON	.COM	8K	MENU FILE TO SEND CODE TO EPSON MX PRINTERS TO SET TYPE SIZE.
XDIR	.COM	4K	EXTENDED DIRECTORY.
SUB	.COM	16K	MENU FILE TO RUN THE MAJOR CP/M COMMAND FILES.
ERASE	.COM	8K	"USER FRIENDLY" ERASE.
SIONS	.COM	12K	FORMATTING PROGRAM FOR BOTH 80 AND 132 COLUMN PRINTERS.
SIONSII	.COM	12K	
BIONS	.TXT	4K	
BIONS	.DOC	4K	
SIONS6	.COM	12K	
FONT	.DAT	4K	
DEL8R	.CDN	16K	TO EXTRACT .L8R FILES TYPE DEL8R FILENAME.
DELBRII	.COM	16K	EXTRACTS .L8R FILES CP/N80, S6 AND MSDOS.
FTNOTE13	.COM	16K	PRODUCES FOOTNOTES WITH WORDSTAR.
FTNOTE13	.DOC	24K	INSTRUCTIONS.
QKI2	.CDN	4K	REDEFINES KEYBOARD.
QKI2	.SUB	4K	" "
QKI2	.DOC	12K	DOCUMENTATION.
BASFK	.ASM	12K	ROUTINE TO LOAD CIFER VDU FUNCTION KEYS WITH BASIC STATEMENTS.
CAT	.CDN	4K	CATALOGUE SYSTEM.
CAT2	.COM	4K	" "
CRCK	.CDN	4K	CHECKSUM PROGRAM.
CRCKLIST	.CRC	4K	CHECKSUM OF SOME FILES ON THIS DISK.
DDISK	.COM	8K	IMPROVED DISK DEBUGGER
DDISK	.MAC	36K	SOURCE OF THE ABOVE
MAST	.CAT	4K	SAMPLE CATALOGUE FILE.
PRTMT/21	.COM	4K	PRINT LISTINGS WITH DATE AND TIME.
PRTMT/21	.ASM	32K	SOURCE OF ABOVE.
PRTMT	.DOC	4K	DOCUMENTATION.
PWS/5	.ABM	32K	WORDSTAR PATCHER FOR INTELLIGENT TERMINALS/PRINTERS
PWS	.DOC	4K	INSTRUCTIONS FOR ABOVE.
TESTZ80	.ZSM	4K	TEST SOURCE FILE FOR Z80 ASSEMBLER.
Z80ASMUK	.ABM	60K	IMPROVED Z80 ASSEMBLER.
Z80ASMUK	.COM	12K	" " "
Z80DOCUK	.DOC	8K	DOC FOR ASSEMBLER.
UDCAT	.CDN	12K	IMPROVED DISK CATALOGUE PROGRAM.
UDCAT	.MAC	20K	
UDCAT	.DOC	8K	DOCUMENTATION FOR ABOVE.

DUTIL.COM 12K DISK UTILITY

A very powerful disk debbugger that lets you view and alter any on disk data found by sector and track number. You can recover erased files by altering the disk directory, dump damaged areas and rewrite them back to disk, examine program code "on-disk", you can even inspect and alter the operating system tracks, perhaps to alter the loader code. Dutil can provide a disk map to indicate which sector and tracks are used by a particular file, this can let you move the good sectors from an unreadable file so that you only have to recreate the one duff sector again. It is a good idea, at first usage, to switch on the printer with CTRL P before entering Dutil, this will produce a hardcopy of the help menus for later reference.

***

QKI2.COM 4K REDEFINES KEYBOARD.
QKI2.SUB 4K " "
QKI2.DOC 12K DOCUMENTATION.

A Keyboard Redefinition Program by Tony Fleig.
QwikKey allows the user to assign character strings to keys. When a key having a string defined in this way is struck, the defined string, rather than the character normally associated with the key, is delivered to the program running at the time. Key definitions may be loaded from files containing previously saved definitions, or they may be defined on-the-fly, even while a program is running. Both normal keys (i.e. keys generating a single character) and keys generating escape sequences are supported. The maximum length of the defined string is 31 minus the length of the character or string normally generated by the key in question. Thirty-one different keys may be defined.

***

RESTORE.COM 4K RESTORES ERASED FILES.

Will recover an ERAsed file. This will only work if the file data has not been overwritten. ERA only alters the first byte of the directory to erase a file. If the first byte of the directory entry is E5 (hex), the file is considered to be erased. RESTORE will zero that first byte. If the file data is undamaged all will be ok, if the file records have been reused since the ERA then chaos will result. It is best to only use Restore if you are sure that the disk has not been written to since the ERA was issued.

***

XDIR.COM 4K EXTENDED DIRECTORY.

Produces a better directory listing than DIR. Arranged in A-Z order with file sizes. Type XDIR or XDIR B: ect.


BLANK
FOR LATER USE